⑲日本国特許庁(JP)　⑪特許出願公開

⑫公開特許公報(A)　平4−60749

⑤Int. Cl.⁵	識別記号	庁内整理番号
G 06 F 15/16	430	9190−5L
9/32	310 J	9189−5B

⑬公開　平成4年(1992)2月26日

審査請求　未請求　請求項の数 2　(全47頁)

⑭発明の名称　デジタルマイクロコンピュータ

⑳特　願　平2−170166
㉒出　願　平2(1990)6月29日

⑫発明者　太期　広一郎　東京都西多摩郡羽村町栄町3丁目2番1号　カシオ計算機株式会社羽村技術センター内
⑫発明者　宇佐美　隆二　東京都西多摩郡羽村町栄町3丁目2番1号　カシオ計算機株式会社羽村技術センター内
⑫発明者　斯波　康祐　東京都西多摩郡羽村町栄町3丁目2番1号　カシオ計算機株式会社羽村技術センター内
⑫発明者　小倉　和夫　東京都西多摩郡羽村町栄町3丁目2番1号　カシオ計算機株式会社羽村技術センター内
⑪出願人　カシオ計算機株式会社　東京都新宿区西新宿2丁目6番1号
⑭代理人　弁理士　杉村　次郎

最終頁に続く

明　細　書

1、発明の名称

デジタルマイクロコンピュータ

2、特許請求の範囲

(1) 第1のＣＰＵと第2のＣＰＵとを有するデジタルマイクロコンピュータにおいて、

前記第1のＣＰＵは前記第2のＣＰＵの内部メモリをアクセスする命令を実行するときにアクセス要求信号を発生するアクセス要求信号発生手段を有し、

前記第2のＣＰＵは前記アクセス要求信号に応答して前記第2のＣＰＵで実行中のオペレーションを中断させ、前記第2のＣＰＵの前記内部メモリを前記第1のＣＰＵのために開放するオペレーション中断手段を有し、

前記第1のＣＰＵは前記オペレーション中断手段により前記内部メモリが前記第1のＣＰＵのために開放されているときに前記内部メモリに対するアクセスのオペレーションを実行するアクセス実行手段を有し、

前記第1のＣＰＵは前記アクセス実行手段による前記アクセスのオペレーションが終了したときにアクセス終了信号を発生するアクセス終了信号発生手段を有し、

前記第2のＣＰＵは前記アクセス終了信号に応答して中断されたオペレーションを再開するオペレーション再開手段を有する

ことを特徴とするデジタルマイクロコンピュータ。

(2) 第1のＣＰＵと第2のＣＰＵとを有するデジタルマイクロコンピュータにおいて、

前記第1のＣＰＵを動作させるクロック信号を発生する第1のクロック発生手段と、

前記第2のＣＰＵを動作させるクロック信号を発生する第2のクロック発生手段と、

前記第1のＣＰＵが前記第2のＣＰＵの内部メモリをアクセスする命令を実行するときに前記第1のＣＰＵから出力されるアクセス要求信号に応

答して前記第２のクロック発生手段の動作を停止することにより、前記第２のＣＰＵで実行中のオペレーションを中断させるクロック停止手段と、

前記アクセス要求信号に応答して前記第２のＣＰＵの前記内部メモリに対するアクセス径路を前記第２のＣＰＵが使用する第１のアクセス径路から前記第１のＣＰＵが使用する第２のアクセス径路に切り換えるアクセス径路切換手段と、

前記アクセス径路切換手段によって切り換えられた前記第２のアクセス径路を介して前記第１のＣＰＵからの前記内部メモリに対するアクセスのオペレーションを実行するアクセス実行手段と、

前記第１のＣＰＵから出力される前記アクセスのオペレーションの終了を表わすアクセス終了信号に応答して、前記内部メモリに対するアクセス径路を前記第２のＣＰＵが使用する前記第１のアクセス径路に復帰させるアクセス径路復帰手段と、

前記アクセス終了信号に応答して前記第２のクロック発生手段の動作を再起動することにより前記第２のＣＰＵで中断されたオペレーションを再開させるクロック再起動手段と、

を有することを特徴するデジタルマイクロコンピュータ。

３．発明の詳細な説明

［発明の技術分野］

この発明はデジタルマイクロコンピュータに関し、特に並列動作可能な複数のＣＰＵを有するデジタルマイクロコンピュータに関する。

［発明の背景］

複数のＣＰＵを有するデジタルマイクロコンピュータは既知である。一般に、ＣＰＵの複数化はコンピュータシステムの性能を上げることをねらいとしている。残念ながら、ＣＰＵの個数をＮ個にしたからといって、システムの性能をＣＰＵが１個の場合のＮ倍に上げることはできない。

複数のＣＰＵのデジタルマイクロコンピュータの性能を低下させる原因の１つとして、ＣＰＵ間のアクセスの問題がある。

従来においてＣＰＵ間のアクセスは次のようにして行われる。まず、アクセスを要求するＣＰＵが要求されるＣＰＵに対し、アクセス要求信号を送る。これに対し、アクセス要求信号を受けたＣＰＵは実行中のオペレーションを完了し、しかる後、内部のメモリを外部の装置（この場合、アクセスを要求したＣＰＵ）がアクセス可能な状況に開放するとともに、アクセスを要求したＣＰＵに対し承認信号を送って停止状態に移行する。アクセスを要求したＣＰＵはこの承認信号を受けた後、相手のＣＰＵの内部メモリに対する実際のアクセスオペレーションを実行する。

したがって、従来のＣＰＵ間のアクセス方式はアクセスの手続に時間がかかる問題がある。特に、ＣＰＵ間で高い頻度のアクセスが要求されるような環境では、アクセス処理がネックとなってＣＰＵの動作効率が著しく低下してしまう。

［発明の目的］

したがってこの発明の目的はＣＰＵ間のアクセスが高速に行えるようにした複数のＣＰＵを有するデジタルマイクロコンピュータを提供することである。

［発明の構成、作用］

この発明によれば、第１のＣＰＵと第２のＣＰＵとを有するデジタルマイクロコンピュータにおいて、前記第１のＣＰＵは前記第２のＣＰＵの内部メモリをアクセスする命令を実行するときにアクセス要求信号を発生するアクセス要求信号発生手段を有し、前記第２のＣＰＵは前記アクセス要求信号に応答して前記第２のＣＰＵで実行中のオペレーションを中断させ、前記第２のＣＰＵの前記内部メモリを前記第１のＣＰＵのために開放するオペレーション中断手段を有し、前記第１のＣＰＵは前記オペレーション中断手段により前記内部メモリが前記第１のＣＰＵのために開放されているときに前記内部メモリに対するアクセスのオ

ペレーションを実行するアクセス実行手段を有し、前記第１のＣＰＵは前記アクセス実行手段による前記アクセスのオペレーションが終了したときにアクセス終了信号を発生するアクセス終了信号発生手段を有し、前記第２のＣＰＵは前記アクセス終了信号に応答して中断されたオペレーションを再開するオペレーション再開手段を有することを特徴とするデジタルマイクロコンピュータが提供される。

この構成によれば、ＣＰＵ間のアクセスは、アクセスを要求するＣＰＵがアクセスが要求されるＣＰＵの内部メモリをアクセスする命令を実行するだけで達成される。したがって、ＣＰＵ間のアクセス処理が高速化され、アクセスを要求するＣＰＵにおける高い動作効率を保証できる。また、アクセスされるＣＰＵも、実際にアクセスが行われている間だけ動作を中断すればよいので動作効率の低下を最小限に抑えることができる。

好ましい構成例において、第１のＣＰＵと第２のＣＰＵとを有するデジタルマイクロコンピュータにおいて、前記第１のＣＰＵを動作させるクロック信号を発生する第１のクロック発生手段と、前記第２のＣＰＵを動作させるクロック信号を発生する第２のクロック発生手段と、前記第１のＣＰＵが前記第２のＣＰＵの内部メモリをアクセスする命令を実行するときに前記第１のＣＰＵから出力されるアクセス要求信号に応答して前記第２のクロック発生手段の動作を停止することにより、前記第２のＣＰＵで実行中のオペレーションを中断させるクロック停止手段と、前記アクセス要求信号に応答して前記第２のＣＰＵの前記内部メモリに対するアクセス経路を前記第２のＣＰＵが使用する第１のアクセス経路から前記第１のＣＰＵが使用する第２のアクセス経路に切り換えるアクセス経路切換手段と、前記アクセス経路切換手段によって切り換えられた前記第２のアクセス経路を介して前記第１のＣＰＵからの前記内部メモリに対するアクセスのオペレーションを実行するアクセス実行手段と、前記第１のＣＰＵから出力される前記アクセスのオペレーションの終了を表わすアクセス終了信号に応答して、前記内部メモリに対するアクセス経路を前記第２のＣＰＵが使用する前記第１のアクセス経路に復帰させるアクセス経路復帰手段と、前記アクセス終了信号に応答して前記第２のクロック発生手段の動作を再起動することにより前記第２のＣＰＵで中断されたオペレーションを再開させるクロック再起動手段とを有することを特徴するデジタルマイクロコンピュータが提供される。

【実施例】

以下、図面を参照してこの発明の実施例を説明する。

＜概　要＞

本実施例はこの発明を電子楽器に適用したものである。本実施例（第１～第３４図）は種々の特徴を含んでいる。第１の特徴は、楽音信号を生成する音源としてプログラムで動作する複数のマイクロコンピュータ処理装置（ＣＰＵ）を使用することであり、従来のような専用構造のハードウェア音源は不要である。１つのＣＰＵがメインＣＰＵあるいはマスターＣＰＵ（１０）として働き、音源処理のみでなくアプリケーション（この場合、楽器）に従う入力装置（鍵盤、機能キー等）、出力装置（ＤＡＣ等）を取り扱う（第４図、第５図）。他のＣＰＵはマスターＣＰＵに対してサブＣＰＵないしスレーブＣＰＵ（２０）として働き、音源処理を実行する（第６図）。したがって、音源処理について各ＣＰＵの負担が分担される構成である。

第２の特徴はサブＣＰＵが動作を開始し、終了するメカニズムに関係しており、本実施例によれば、サブＣＰＵの動作は、マスターＣＰＵに対して音源処理を要求するタイマインタラプトを合図として開始し、その結果、マスターＣＰＵとサブＣＰＵにおいて音源処理が並行に実行される。サブＣＰＵの動作（音源処理）が終了するとその終了信号によってサブＣＰＵはリセット状態（停止状態）に移行するとともにその終了信号がマスタ

ーＣＰＵに伝えられる（第８図、第１６図）。この特徴により、マスターＣＰＵはサブＣＰＵの動作期間を有効に管理、把握できる。更に、この特徴により、高速処理が要求される音源処理タスク（楽音信号のデジタルサンプルを生成する仕事）を効率よく実行できる。

本実施例の第３の特徴はメインプログラムからタイマインタラプト処理ルーチンに渡すデータの更新（転送）問題に関係する。インタラプト処理ルーチンの実行の結果、インタラプト処理ルーチンにおいて参照すべき複数のデータ（例えばエンベロープ目標値、エンベロープレートのようなエンベロープパラメータ）を更新する必要が生じる。この複数のデータの更新の実行命令はメインプログラム中に含まれる。即ち、この複数のデータはメインプログラムが更新し、タイマインタラプト処理ルーチンが参照するデータである。このような複数のデータは、全体として意味ある情報を構成するので、メインプログラムにおいて複数のデータのすべてが更新されないうちにインタラプト処理ルーチンに制御が移ってはならない。これを防止するため、第１の方式としてデータ更新が完了するまでインタラプトをマスクしてインタラプト処理ルーチンへの移行を禁止する方式が開示され（第１６図、第１７図）、第２の方式として、複数のデータの更新（転送）をメインプログラム中の単一命令で実行する方式が開示される（第１８図～第２１図）。この結果、インタラプト処理ルーチンの処理結果（楽音信号のサンプル）が正しい値を示し、正しい動作が保証される。

本実施例の第４の特徴はマスターＣＰＵからスレーブＣＰＵに対するデータアクセス問題に関する。従来の複数ＣＰＵマイクロコンピュータシステムでは、一般に、ＣＰＵ間のデータ転送は一連のシーケンスを通して行われ、相当の時間を要する。代表的には、データのアクセスを要求するＣＰＵからアクセスが要求されるＣＰＵに対し、アクセス要求信号を送る。このアクセス要求信号に対しアクセスが要求されるＣＰＵは実行中のオペレーションを完了した後に承認（アクノリッジ）信号をＣＰＵに渡して停止状態となる。アクセス要求信号送信後、承認信号が受信されるまでの間、要求側のＣＰＵは待ち状態になる。承認信号を受けて要求側のＣＰＵは被要求側のＣＰＵの内部メモリに対し、実際のデータアクセスを実行する。このように従来のＣＰＵ間データアクセス方式は時間を要するので高速処理が望まれる電子楽器のようなアプリケーションには適さない。これを解決するため、本実施例では、第１のデータアクセス方式として、上記第２の特徴を利用してサブＣＰＵが停止状態にあるときにマスターＣＰＵがサブＣＰＵの内部メモリ（２０６）に対しデータをリード／ライト（アクセス）する停止モード制御方式が開示され（第２２図）、第２のデータアクセス方式として待ち状態なしにマスターＣＰＵがサブＣＰＵをデータアクセスする（サブＣＰＵはデータアクセス中のみ強制的に停止状態にされる）割時データアクセス方式とが開示される（第２３図～第２５図）。

本実施例の第５の特徴はデータ源としてのＣＰＵ外部メモリを複数のＣＰＵで共用する場合における複数ＣＰＵからのアクセスの競合（衝突）問題に関する。本実施例によれば後述するメモリ装置競合回避回路（５０）を設けることにより、共用メモリに対するアクセスの競合を解消し、一定の待ち時間の後、共用メモリからのデータを得られるようにしている。

本実施例の第６の特徴はデータ変換処理（シフト、反転、一部取り出し等）の高速化に関する。従来においては、上述したＣＰＵ外部メモリのようなデータメモリ内のデータから、ＣＰＵ内部メモリ（演算用メモリ）上に変換されたデータを得るために、転送（リードアクセス）命令により、データメモリのデータを演算用メモリに移し、しかる後、変換命令により、演算用メモリのデータをＡＬＵを介して変換する。所望のデータ変換を行うために複数の変換命令を実行する必要もしばしば生じる。このように、従来においてはデータ変換の処理に時間がかかるという問題があり、特

に音源処理のように高速処理が要求されるアプリケーションにおいては大きな問題となる。これを解決するため、この実施例によれば、データ・アドレス変換ハードウェア（６０、７０）を設け、特殊な転送命令（変換付転送命令）を実行することにより、その命令に応答するデータ・アドレス変換ハードウェアを介して所望のデータ変換が施されたデータが演算用メモリ（１０６、２０６）に取り込まれるようにしている。したがって、所望の変換データを得るのに、複数の命令を実行するのではなく単一の命令を実行すればよく処理の高速化が図れる。

＜全体構成（第１図）＞

第１図は電子楽器の処理装置として構成した本実施例の全体構成を示すブロック図である。本システムは２つの中央演算処理装置（一方をＭＣＰＵ１０、他方をＳＣＰＵ２０で示す）を有する。各ＣＰＵ１０、２０はプログラムを内蔵しており、それぞれのプログラムに従って動作する。ＭＣＰＵ１０は音源処理（第５図）以外にシステム全体の制御、例えば入力ポート１１８、出力ポート１２０に接続される入力装置（例えば鍵盤、機能キー等）からの入力情報の処理、デジタル楽音信号をアナログ楽音信号に変換するＤＡＣ１００の制御等を行う（第４図）。これに対し、ＳＣＰＵ２０は音源処理に専用される（第６図）。

９０は音源制御データ、波形データ等のデータ源としてのメモリである。データメモリ９０はここでは、ＬＳＩチップ（第１図の残りのデバイスを搭載している）に外付けされたＲＯＭで構成されている。集積度が高ければ、単一のＬＳＩチップ上にデータメモリ９０を内部メモリとして形成可能である。外部メモリ９０はＭＣＰＵ１０とＳＣＰＵ２０に共用される。ＭＣＰＵ１０からのアドレス情報はＭＣＰＵ１０に結合するアドレスバスＭＡ、外部メモリアドレスラッチ３０のＭＣＰＵ外部メモリアドレスラッチ３０Ｍ、アドレス切り換え回路４０、アドレス変換回路６０を介して外部データメモリ９０のアドレス入力に加えられる。一方、ＳＣＰＵ２０からのアドレス情報はＳＣＰＵ２０に結合するアドレスバスＳＡ、ＳＣＰＵ外部メモリアドレスラッチ３０Ｓ、アドレス切り換え回路４０、アドレス変換回路６０を通して外部データメモリ９０のアドレス入力に加えられる。外部データメモリ９０からＭＣＰＵ１０へのデータ伝送径路は外部データメモリ９０のデータ出力、データ変換回路７０、外部メモリデータラッチ８０のＭＣＰＵ外部メモリデータラッチ８０Ｍ、ＭＣＰＵ１０に結合するデータバスＭＤによって構成される。これに対し、外部データメモリ９０からＳＣＰＵ２０へのデータ伝送径路は外部データメモリ９０のデータ出力、データ変換回路７０、ＳＣＰＵ外部メモリデータラッチ８０Ｓ、ＳＣＰＵ２０に結合するデータバスＳＤによって構成される。

メモリ装置競合回避回路５０はＭＣＰＵ１０とＳＣＰＵ２０の両ＣＰＵによる外部メモリ９０のアクセスを制御し、その競合を回避するものである。メモリ装置競合回避回路５０はＭＣＰＵ１０からの外部メモリアクセスを要求する信号ｒｏｍａとＳＣＰＵからの外部メモリアクセスを要求する信号ｒｏｍａの各々に応答してアドレス切り換え回路４０を制御してアドレス切り換え回路４０にＭＣＰＵ１０からのアドレスとＳＣＰＵ２０からのアドレスのいずれかを外部メモリ９０へのアドレスとして選択させる。このためにメモリ装置競合回避回路５０からの選択信号ＭＳＥＬによりアドレス切り換え回路４０は選択動作を行う。外部メモリ９０へのアドレスが確定するとメモリ装置競合回避回路５０は外部メモリ９０に対するチップ選択信号ＣＥと出力イネーブル信号ＯＥをアクティブにする。これにより外部メモリ９０からデータが出力され、データ変換回路７０を通してそのデータが外部メモリラッチ８０の入力バスに現われる。ここで、メモリ装置競合回避回路５０はデータアクセスを要求したＣＰＵにデータを送るためにＭＣＰＵ外部メモリデータラッチ８０Ｍ、ＳＣＰＵ外部メモリデータラッチ８０Ｓのいずれかを作動してデータをラッチさせる。このた

めにＭＣＰＵ外部メモリデータラッチ８０Ｍはメモリ装置競合回避回路５０からのラッチ信号ＭＤＬによりラッチ動作し、ＳＣＰＵ外部メモリデータラッチ８０Ｓはメモリ装置競合回避回路５０からのラッチ信号ＳＤＬによりラッチ動作するようになっている。

アドレス変換回路６０とデータ変換回路７０は外部データメモリ９０のデータを変換したデータがＣＰＵ１０、２０に取り込まれるようにするための変換デバイスである。アドレス変換回路６０はアドレス切り換え回路４０を通ったアドレス、即ち、ＣＰＵ（ＭＣＰＵ１０かＳＣＰＵ２０）から出力されたアドレス（論理アドレス）を選択的に変更して外部データメモリ９０に実際に入力されるアドレスを形成するものであり、データ変換回路７０は外部データメモリ９０から出力されたデータを選択的に変更してＣＰＵ（ＭＣＰＵ１０かＳＣＰＵ２０）に実際に入力されるデータを形成するものである。各変換回路６０、７０における変換の態様を指定するために、制御信号が使用される。各ＣＰＵ１０、２０において、外部データメモリ９０に対するデータアクセスは転送命令を実行することで行われる。転送命令に基づいてＣＰＵで生成される制御信号をＭＲ１、ＭＲ２、ＭＲ３（ＭＣＰＵ１０の場合）、ＳＲ１、ＳＲ２、ＳＲ３（ＳＣＰＵ２０の場合）で示してある。これらの信号は外部メモリアドレスラッチ３０、アドレス切り換え回路４０を通った後、信号Ｒ１、Ｒ２、Ｒ３と呼ばれる（ＭＲｉ→ＬＭＲｉ→ＲｉまたはＳＲｉ→ＬＳＲｉ→Ｒｉ）。変換の態様を指定するため、制御信号Ｒ１、Ｒ２がアドレス変換回路６０に入力される。更に、データ変換回路７０における変換の態様を特定するため、制御信号Ｒ１、Ｒ２、Ｒ３とアドレス変換回路６０からのアドレスビット１２の信号Ａ１２とアドレスビット１５の信号Ａ１５がデータ変換回路７０に加えられる。アドレス変換回路６０とデータ変換回路７０の詳細については後述する。

ＭＣＰＵ１０とＳＣＰＵ２０との間のインタフェースを定めるため、両ＣＰＵ間で複数の信号が伝送される。信号ＡはＭＣＰＵ１０からＳＣＰＵ２０に送られるＳＣＰＵ２０の処理開始を表わす信号、信号ＢはＳＣＰＵ２０からＭＣＰＵ１０に送られるＳＣＰＵ２０の処理終了を表わす信号、ＭａはＭＣＰＵ１０からＳＣＰＵ２０に送られるＳＣＰＵ２０の内部メモリ（第３図の２０６）のアドレス情報、信号ＣはＭＣＰＵ１０からＳＣＰＵ２０に送られるＳＣＰＵ２０の内部メモリの読み書き制御信号、ＤｉｎはＳＣＰＵ２０からＭＣＰＵ１０に送られるＳＣＰＵ２０の内部メモリからの読出しデータ、Ｄｏｕｔ はＭＣＰＵ１０からＳＣＰＵ２０に送られるＳＣＰＵ２０の内部メモリへの書き込みデータを表わす、ＣＰＵ間インタフェースの詳細については後述する。

上述したように音源処理によりＭＣＰＵ１０とＳＣＰＵ２０とでデジタル楽音信号が生成される。生成結果はＭＣＰＵ１０から、右ＤＡＣ１００Ｒと左ＤＡＣ１００Ｌとから成るデジタルアナログ変換器（ＤＡＣ）１００に送られ、アナログ楽音信号に変換されて外部に出力される。

＜ＭＣＰＵとＳＣＰＵの構成（第２、第３図）＞

第２図にＭＣＰＵ１０の内部構造を示し、第３図にＳＣＰＵ２０の内部構造を示す。

第２図において制御用ＲＯＭ１０２には楽器の各種制御入力を処理するメインプログラムと楽音を生成するインタラプト処理プログラムが記憶されており、ＲＯＭアドレス制御部１１４からＲＯＭアドレスデコーダ１０４を介して指定されたアドレスにあるプログラム語（命令）をインストラクション出力ラッチ１０２ａを介して順次出力していく。なお、具体的実施例では、プログラム語長は２８ビットであり、プログラム語の一部が次に読み出されるべきプログラム語を記憶するアドレスの下位部（ページ内アドレス）としてＲＯＭアドレス制御部１１４に入力されるネクストアドレス方式となっているが、代りにプログラムカウンタ方式を使用してもよい。ＲＡＭアドレス制御部１１４は制御用ＲＯＭ１０２からの命令のオペランドがレジスタを指定している場合に、ＲＡＭ１０６内の対応するレジスタのアドレスを指定す

る。ＲＡＭ１０５は演算用メモリを構成するレジスタ群であり、汎用演算、フラグ演算、楽音の演算等に使用される。ＡＬＵ部（加減算器及び論理演算部）１０８と乗算器１１０は制御用ＲＯＭ１０２からの命令が演算命令のときに用いられる。特に乗算器１１０は楽音波形の演算に使用しており、そのための最適化として第１と第２のデータ入力（例えば１６ビットデータ）を乗算して入力と同じ長さ（１６ビット）のデータを出力するようになっている。上記ＲＡＭ１０５、加減算器１０８、乗算器１１０により、演算回路が構成される。オペレーション制御回路１１２は制御用ＲＯＭ１０２からの命令のオペコードを解読し、指示されるオペレーションを実行するために、回路の各部に制御信号（全体をＣＮＴＲで示す）を送る。また条件付分岐命令の実行の際にオペレーション制御回路１１２はＡＬＵ部１０８からのステータス信号Ｓ（例えばオーバーフロー信号、ゼロフラグ信号等）により分岐条件成立を検出してＲＯＭアドレス制御部１１４を介してアドレスを分岐先のアドレスにジャンプさせる。

所定時間ごとに制御用ＲＯＭ１０２の楽音生成プログラムを実行するため、この実施例ではタイマインタラプトを採用している。すなわち、タイマ（ハードウェアカウンタ）を有するインタラプト発生部１１６により、一定時間ごとにＲＯＭアドレス制御部１１４に制御信号ＩＮＴ（割込要求信号）を送り、この信号により、ＲＯＭアドレス制御部１１４は次に行うメインプログラムの命令のアドレスを退避（保持）し、楽音の生成が行われるインタラプト処理プログラム（サブルーチン）の先頭アドレスを代りにセットする。これにより、インタラプト処理プログラムが開始される。インタラプト処理プログラムの最後にはリターン命令があるので、このリターン命令がオペレーション制御回路１１２で解読された時点で、ＲＯＭアドレス制御部１１４は退避してあったアドレスを再度セットし、メインプログラムに復帰する。更に、インタラプト発生部１１６からの制御信号ＩＮＴはＤＡＣ１００における楽音信号のデジタル／アナログ変換サンプリング速度を定めるためにＤＡＣ１００に供給される。なお、インタラプト発生部１１６は図の上ではＭＣＰＵ１０の内部要素として描いてあるが、ＭＣＰＵ１０に対して現在行っている仕事を停止させ特別の処理を要求するものであり、論理的にはＭＣＰＵ１０の外部要素（周辺装置）である。

クロック発生回路１３６はマスタークロック発生回路（図示せず）からの２相のマスタークロックＣＫ１とＣＫ２を受け、オペレーション制御回路１１２を初めとする回路の各部に加える種々のタイミング信号（Ｔ１、Ｔ２、Ｔ３、Ｔ１ＣＫ１、Ｔ２ＣＫ２、Ｔ３ＣＫ３等）を発生する。

第２図の残りの要素はＭＣＰＵ２０の外部装置とのインタフェースに係っている。１２２は外部メモリアクセス用アドレスバスＭＡ（第１図）にＭＣＰＵ内部バスを接続するためのバスインタフェースとしてのゲートを表わし、１２４は外部メモリデータバスＭＤにＭＣＰＵ内部バスを接続するためのゲートを表わし、１２６はＤＡＣデータ転送バスにＭＣＰＵ内部バスを接続するためのゲートを表わす。また、入力ポート１１８と出力ポート１２０はＭＣＰＵ内部バスを外部の入力装置に結合するためのインタフェースである。１２８はＳＣＰＵ内部ＲＡＭアドレス指定バスにＭＣＰＵ内部バスを接続するためのゲート、１３０はＳＣＰＵ内部ＲＡＭ書込データバスにＭＣＰＵ内部バスを接続するためのゲート、１３２はＳＣＰＵ内部ＲＡＭ読出データバスをＭＣＰＵ内部バスに接続するためのゲートを表わす。

ＳＣＰＵリセット制御部１３４はＳＣＰＵ２０の動作期間を管理するためのデバイスである。この実施例に従いＳＣＰＵリセット制御部１３４はインタラプト発生部１１５からのインタラプト信号ＩＮＴに応答して、ＳＣＰＵ２０の処理開始を示す信号Ａを発生する。この信号ＡはＳＣＰＵ２０のＲＯＭアドレス制御部２１４（第３図）に送られ、これによりＲＯＭアドレス制御部２１４のアドレス更新動作が開始し、ＳＣＰＵ２０の動作（音源処理を含む）が開始する。ＳＣＰＵ２０の

動作が終了するとＳＣＰＵ２０のオペレーション制御回路２１２から処理終了を示す信号Ｂが発生し、この信号ＢがＳＣＰＵリセット制御部１３４に送られる。これに対し、ＳＣＰＵリセット制御部１３４はＳＣＰＵ２０の動作を停止するために信号Ａを反転し、これによりＳＣＰＵ２０のＲＯＭアドレス制御部２１４の動作を停止させる、とともに、ＳＣＰＵ２０が停止中であることを表わすＳＣＰＵ状態フラグ信号をオペレーション制御回路１１２に送る。オペレーション制御回路１１２は制御用ＲＯＭ１０２からのＳＣＰＵ状態の検査命令の実行時に、このＳＣＰＵ状態フラグ信号を読むことにより、ＳＣＰＵ２０の状態を検出できる。

第３図のＳＣＰＵ２０のブロック図において、要素２０２、２０２ａ、２０４、２０５、２０６、２０８、２１２、２１４、２２２、２２４、２３６はそれぞれ、第２図のＭＣＰＵ１０のブロック図における要素１０２、１０２ａ、１０４、１０５、１０６、１０８、１１０、１１２、１１４、１２２、１２４、１３６に対応する要素である。ただし、ＳＣＰＵ２０の制御用ＲＯＭ２０２には基本的に音源処理のためのプログラムのみが記憶されており、ＳＣＰＵ２０を音源処理専用の処理装置として機能させている。

２４０はＳＣＰＵ２０の演算用メモリとしてのＲＡＭ２０６へ入力するデータをＭＣＰＵ１０からのデータ（ＭＣＰＵ１０からゲート１３０、データバスＤｏｕｔを通ったデータ）とＳＣＰＵ２０の生成（演算）したデータ（ＡＬＵ部２０８または乗算器２１０からのデータバスＤＢ上のデータ）とから選択するＲＡＭデータイン切り換え部である。ＲＡＭデータイン切り換え部２４０は信号Ａによってその選択モードが制御され、信号Ａが“ＳＣＰＵ２０動作中”を表わしているときにはＳＣＰＵ２０で演算したデータを選択し、信号Ａが“ＳＣＰＵ２０停止中”を表わしているときにはＭＣＰＵ１０からのデータを選択する。

また、ＲＡＭアドレス制御部２０５も、信号Ａによってそのモードが制御され、信号Ａが“ＳＣＰＵ２０動作中”を表わしているときには制御用ＲＯＭのインストラクション出力ラッチ２０２ａからのバスＳＡ上の情報をＲＡＭ２０６のアドレスとして選択し、信号Ａが“ＳＣＰＵ２０停止中”を表わしているときにはＭＣＰＵ１０からバスゲート１２８（信号Ａにより開いている）を経てバスＭａ上にあるＭＣＰＵ１０からの情報をＲＡＭ２０６のアドレスとして選択する。同様に、ライト信号切り換え部２４２も信号Ａによってそのモードが制御され、信号Ａが“ＳＣＰＵ２０動作中”を表わしているときにはＳＣＰＵ２０のオペレーション制御回路２１２からのＲＡＭリードライト信号を選択してＲＡＭ２０６のリードライト入力$\overline{R}$／Ｗに結合し、信号Ａが“ＳＣＰＵ２０停止中”を表わしているときにはＳＣＰＵ２０ではなくＭＣＰＵ１０のオペレーション制御回路１１２からのＳＣＰＵＲＡＭリードライト信号を選択してＲＡＭ２０６のリードライト入力$\overline{R}$／Ｗに結合する。

以下、本実施例の諸特徴を更に詳細に説明する。

＜複数ＣＰＵ音源機能（第１～第７図、第９～第１１図）＞

第４図はＭＣＰＵ１０のメインプログラム（バックグランドプログラム）によるＭＣＰＵ１０の動作を示すフローチャート、第５図はタイマインタラプト信号ＩＮＴによって起動されるＭＣＰＵ１０のインタラプト処理ルーチンによるＭＣＰＵ１０の動作を示すフローチャート、第６図はタイマインタラプト信号ＩＮＴによって起動されるＳＣＰＵ２０のプログラムによるＳＣＰＵ２０の動作を示すフローチャート、第７図はＭＣＰＵ１０とＳＣＰＵ２０のそれぞれが実行する音源処理のフローチャートである。

第１～第３図に関して述べたように、本実施例の電子楽器処理システムはＭＣＰＵ１０とＳＣＰＵ２０とから成る複数のＣＰＵを備えており、両ＣＰＵが協働して電子楽器のための処理を実行する。特にＭＣＰＵ１０は、第５図に示すようなイ

ンタラプト処理ルーチンにより音源処理を行い、ＳＣＰＵ２０は第６図に示すようなプログラムにより音源処理を行う。更にＭＣＰＵ１０は第４図に示すメインプログラムにより、システム全体の制御のための種々のタスクを実行する。

第４図のメインプログラムのフローにおいて、４－１は電源投入時にシステムを初期化する処理であり、ＭＣＰＵ１０はＲＡＭ１０６、ＲＡＭ２０６のクリアや、リズムテンポ等の初期値の設定等を行う。４－２でＭＣＰＵ１０は出力ポート１２０からキー走査のための信号を出力し、鍵盤、機能スイッチ等の入力装置の状態を入力ポート１１８から取り込むことにより、機能キー、鍵盤キーの状態をＲＡＭ１０６のキーバッファエリアに記憶する。４－３では４－２で得た機能キーの新しい状態と前回の状態とから、状態の変化した機能キーを識別し、指示される機能の実行を行う（例えば、楽音番号のセット、エンベロープ番号のセット、リズム番号のセット等）。４－４では４－２で得た鍵盤の最新の状態と前回の状態とから、変化した鍵（押鍵、離鍵）を識別する。次の４－５で４－４の処理結果から、発音処理４－９のためのキーアサイン処理を行う。４－６では機能キーでデモ演奏キーが押鍵されたとき外部メモリ９０から、デモ演奏データ（シーケンサデータ）を順次読み出し、処理することにより、発音処理４－９のためのキーアサイン処理等を行う。４－７ではリズムスタートキーが押鍵されたとき外部メモリ９０からリズムデータを順次読み出し、発音処理４－９のためのキーアサイン処理を行う。フロー一周タイマ処理４－８では、メインフローで必要なイベントのタイミングを知るために、フロー一周時間（これは、フローを一周する間に実行されたタイマインタラプトの回数を計数することで得られる。この計数処理は後述のインタラプトタイマ処理５－２で行われる。）を基に演算を行い、エンベロープ用タイマ（エンベロープの演算周期）やリズム用の基準値を得る。発音処理４－９では４－５、４－６、４－７でセットされたデータから、実際に楽音を発音させるための各種演算を行い、結果をＲＡＭ１０６、ＲＡＭ２０６内の音源処理レジスタ（第１１図）にセットする。４－１０は次のメインフローのパスのための準備処理であり、今回のパスで得た押鍵状態への変化を示すＮＥＷ　ＯＮ状態をＯＮ中にしたり、離鍵状態への変化を示すＮＥＷ　ＯＦＦ状態をＯＦＦ中に変える等の処理を行う。

インタラプト発生部１１６からインタラプト信号ＩＮＴが発生すると、ＭＣＰＵ１０は実行中のメインプログラムを中断し、第５図に示すインタラプト処理ルーチンを実行し、ＳＣＰＵ２０は第６図に示すプログラムを実行する。ここにＭＣＰＵ１０は第５図のフローにおいて楽音信号を生成し、ＳＣＰＵ２０は第６図のフローにおいて楽音信号を生成するようになっている。

詳細に述べるとＭＣＰＵ１０は５－１で各チャンネルに対する楽音波形データを生成し、累算し、記憶する。従来はこの処理を音源回路ハードウェアで行っていた。次のインタラプト処理タイマ処理５－２でＭＣＰＵ１０はインタラプトが一定時間ごとにかかることを利用して、フロー一周計時用のタイマレジスタ（ＲＡＭ１０６内）を通過の都度、プラス１する。５－３でＭＣＰＵ１０はＳＣＰＵ２０の音源処理６－１が終了しているかどうかを検査し、終了していれば、５－４に進んで、ＳＣＰＵ２０で生成されたＲＡＭ２０６上の楽音波形データをＲＡＭ１０６内に読み込む。そして５－５でＭＣＰＵ１０はＭＣＰＵ１０の生成した楽音波形データとＳＣＰＵ２０で生成した楽音波形データをＤＡＣ１００に出力する。

音源処理５－１、６－１の詳細を第７図に示す。本例では、各ＣＰＵ（ＭＣＰＵ１０、ＳＣＰＵ２０）はそれぞれ８チャンネル分の楽音波形データを生成可能であり、システム全体として１６チャンネル分の楽音波形データを生成可能としている。７－１で波形加算用ＲＡＭ領域（ＲＡＭ１０６内、ＲＡＭ２０６内）をクリアし、７－２～７－９で第１チャンネルから第８チャンネルまでの各チャンネル音源処理を順次実行する。各チャンネル音源処理の最後で、チャンネルの楽音波形

値が波形加算用ＲＡＭ領域のデータに加算される。

次にチャンネル音源処理の例について第９図～第１１図を参照して説明する。この例では、波形読み出し（ＰＣＭ）方式の楽音合成を採用している（他の楽音合成方式、例えばＦＭ合成も実現可能であり、この発明は特定の楽音合成方式には制限されない）。チャンネル音源処理は大きくわけて、エンベロープ処理（９－１～９－７）と、エンベロープ付加を含む波形処理（９－８～９－２１）とから成る。各ＣＰＵ（ＭＣＰＵ１０、ＳＣＰＵ２０）はチャンネル音源処理を実行する際に、そのチャンネルに対する音源処理レジスタ群、即ち第１１図に示すように、エンベロープΔｘ用タイマー、目標エンベロープ、エンベロープΔｘ、加減フラグ付エンベロープΔｙ、現在エンベロープ、アドレス加算値、ループアドレス、エンドアドレス、スタートアドレス兼現在アドレスを参照し、所望のレジスタを更新する。エンベロープは振幅変調のために基本波形に付加すべきもので、全体としていくつかのセグメント（ステップ）から成っている。エンベロープΔｘ用タイマーと目標エンベロープとエンベロープΔｘと加減フラグ付エンベロープΔｙは現在進行中のエンベロープセグメントを定義するエンベロープパラメータであり、このエンベロープパラメータは、ＭＣＰＵ１０のメインプログラム（第４図）の発音処理４－９内において、エンベロープ値がセグメントの目標値に到達の都度、更新される情報であり、インタラプト処理ルーチン（第５図、第６図）ではこれらのエンベロープパラメータはエンベロープΔｘ用タイマーを除いて単に参照されるだけである。エンベロープΔｘはエンベロープの演算周期を表わし、目標エンベロープは現セグメントにおけるエンベロープの目標値を表わし、加減フラグ付エンベロープΔｙは演算周期ごとのエンベロープの変化分を表わし、現在エンベロープは現在のエンベロープ値を表わす。アドレス加算値、ループアドレス、エンドアドレス及びスタートアドレス兼現在アドレスは外部メモリ９０に置かれる基本波形に対するアドレス情報であり、スタートアドレスは基本波形メモリ（外部メモリ９０内）のスタートアドレス、ループアドレスは基本波形を繰り返し読み出す場合の戻り先のアドレス（第１０図ではスタートアドレスと同一）、エンドアドレスは基本波形のエンドアドレスを表わし、現在アドレスは基本波形の現在の位相を表わすアドレスであり、その整数部が、基本波形メモリに現実に存在する記憶場所を表わし、その小数部が、この記憶場所からのずれを表わし、アドレス加算値はタイマインタラプト処理ルーチンの時間間隔ごとに現在アドレスに加算されるべき値であり、生成する楽音のピッチに正比例する。

詳細に述べると、９－１でエンベロープの演算周期Δｘと比較するためのタイマレジスタをインタラプトごとにインクリメントし、９－２でΔｘと一致したとき９－３でエンベロープ変位分のデータΔｙの加減算フラグ（符号ビット）をテストしてエンベロープが上昇中か下降中かを判別し、９－４、９－５でそれぞれ現在エンベロープの減算または加算を行う。９－６で現在エンベロープが目標エンベロープ値に達したかどうかをチェックし、達しておれば、現在エンベロープに目標レベルをセットする。これによりメインプログラムの発音処理４－９で次のエンベロープステップのデータがセットされることになる。また発音処理４－９でゼロの現在エンベロープを読んだときには発音の終了として処理される。

次に、波形処理９－８～９－２１について述べる。波形処理では、現在アドレスの整数部を使って基本波形メモリから隣り合う２つアドレスの波形データを読み出し、（整数部＋小数部）で示される現在アドレスに対して想定される波形値を補間で求めている。補間が必要な理由は、タイマインタラプトによる波形サンプリング周期が一定であり、アドレスの加算値（ピッチデータ）が楽器への応用上、ある音域にわたるためである（音階音しか出力しない楽器で音階音ごとに波形データを用意すれば補間の必要はないが許容できない記憶容量の増大となる）。補間による音色の劣化、

歪みは高音域の方が著しいため、原音の記録サンプリング周期より高速の周期で原音を再生するのが好ましい。この実施例では原音（４－４）再生の周期を２倍にしている（第１０図）。したがって、アドレス加算値が0.5のとき、Ａ４の音が得られるようになっている。この場合、Ａ＃４ではアドレス加算値は0.529となり、Ａ３のとき、１となる。これらのアドレス加算値はピッチデータとして制御データ兼波形外部メモリ９０内に記憶されており、押鍵時には発音処理４－９において、鍵に対応するピッチデータと選択されている音色の波形スタートアドレス、波形エンドアドレス及び波形ループアドレスがＲＡＭ１０６またはＲＡＭ２０６の対応するレジスタ、すなわち、アドレス加算値レジスタ、スタートアドレス兼現在アドレスレジスタ、エンドアドレスレジスタ、ループアドレスレジスタにセットされる。

参考までに、第１０図に時間に対する補間波形データを示す。図中、白丸は基本波形メモリの記憶場所にある波形データ値、×印は補間値を含む出力サンプルを示している。

補間の方式はいろいろあるが、ここでは直線補間を採用している。詳細に述べると、まず、９－８で現在アドレスにアドレス加算値を加算して新しい現在アドレスを得る。９－９で現在アドレスとエンドアドレスを比較し、現在アドレス＞エンドアドレスならば、９－１０、９－１１により、現在アドレス＜エンドアドレスのときは９－１２により、物理上（番地上）または論理上（動作上）の次のアドレスを計算し、９－１４でその整数部により基本波形メモリをアクセスして次回波形データを得る。ループアドレスは動作上エンドアドレスの次のアドレスである。すなわち、第１０図の場合、図示の波形は繰り返し読み出される。したがって、現在アドレス＝エンドアドレスのときは次のアドレスとしてループアドレスの波形データを読み出す（９－１３）。９－１５、９－１６により、現在アドレスの整数部で基本波形をアクセスして今回の波形データを読み出す。次に、９－１７で次回波形値から今回波形値を減算し、９－１８でその差に現在アドレスの小数部を乗算し、その結果を９－１９で今回の波形値に加えることにより、波形の直線補間値を求める。この直線補間したデータに現在エンベロープ値を乗算してチャンネルの楽音データ値を得（９－２０）、それを波形加算用レジスタの内容に加えて楽音データを累算する（９－２１）。このレジスタに累算されたデジタル楽音データがタイマインタラプト処理ルーチン（第５図）の５－５でＤＡＣ１００に送出される。これに関連し、第１図ではＤＡＣ１００はステレオ出力を得るべく右ＤＡＣ１００Ｒと左ＤＡＣ１００Ｌから成っている。この場合、ＭＣＰＵ１０、ＳＣＰＵ２０の処理する音源チャンネルの夫々を左右のＤＡＣのいずれに割り当てるかを決めるようにするとよい。具体的には、各チャンネル用の音源データとして内部ＲＡＭ１０６、２０６上に、選択ＤＡＣ指示データをもたせ、また、２つの波形加算用領域、即ち、左ＤＡＣ用波形加算用領域と左ＤＡＣ用波形加算用領域を設ける。また、７－１に対応するステップで左右のＤＡＣ用の各波形加算用領域をクリアし、９－２０の処理の後、処理チャンネルに割り当てているＤＡＣを選択ＤＡＣ指示データから判別し、対応する波形加算用領域に処理チャンネルの楽音波形データを加算する。そして、ＭＣＰＵ１０のインタラプト処理ルーチン（第５図）のステップ５－４に対応するステップで、ＳＣＰＵ２０の生成した左ＤＡＣ用楽音波形データと右ＤＡＣ用楽音波形データとをそれぞれＭＣＰＵ１０で生成した左ＤＡＣ用楽音波形データと右ＤＡＣ用楽音波形データに加算し、加算結果である左ＤＡＣ用と右ＤＡＣ用の楽音波形データを５－５に相当するステップで、それぞれ左ＤＡＣ１００Ｌと右ＤＡＣ１００Ｒに送出する。

このように、本実施例の電子楽器用処理装置はＭＣＰＵ１０とＳＣＰＵ２０という複数のＣＰＵを有し、各ＣＰＵにおいて、内蔵されるプログラムに従って音源処理を実行することができる。なお実施例では１つのＳＣＰＵを使用しているが、音源処理を行う複数のＳＣＰＵを設けるようにし

てもよい。

＜ＳＣＰＵ動作開始・終了機能（第１２～第１５図、第２～第６図、第８図）＞

本実施例によればＭＣＰＵ１０はＳＣＰＵ２０の動作期間を管理、把握する機能を有している。この目的のため、

（イ）ＭＣＰＵ１０はタイマ・インタラプト発生部１１６からインタラプト信号が発生したときに、これを合図としてＳＣＰＵ２０の動作を開始させ、ＭＣＰＵ１０のオペレーション制御回路１１２が参照するＳＣＰＵ状態フラグを“ＳＣＰＵ動作中”にセットする。

（ロ）ＳＣＰＵ２０は動作（音源処理）を完了したときに、これに応答して停止状態に移行し、ＭＣＰＵ１０に動作完了信号を送り、ＭＣＰＵ１０のオペレーション制御回路１１２が参照するＳＣＰＵ状態フラグを“ＳＣＰＵ停止中”にセットする。

第２図～第６図を参照すると、ＭＣＰＵ１０はメインプログラム（第４図）の実行中に、インタラプト発生部１１６（第２図）からインタラプト信号を受けると、ＲＯＭアドレス制御部１１４を介してメインプログラムを中断し、楽音生成のために第５図に示すタイマインタラプト処理ルーチンを実行する。更に、ＭＣＰＵ１０はインタラプト信号に対し、ＳＣＰＵリセット制御部１３４を介してＳＣＰＵ２０にＳＣＰＵ動作開始信号Ａを送り、これを受けてＳＣＰＵ２０はＲＯＭアドレス制御部２１４を介して第６図に示す楽音生成のためのプログラムを実行する（なお信号Ａにより、バスゲート１２８、ＲＡＭアドレス制御部２０４、ＲＡＭデータイン切り換え部２４０、ライト信号切り換え部２４２も、ＳＣＰＵ２０自身の動作のためにセットされる）。このプログラムの終了に伴い、ＳＣＰＵ２０はオペレーション制御回路２１２から動作終了信号Ｂを発生する。この信号ＢはＳＣＰＵリセット制御部１３４に送られ、これを受けてＳＣＰＵリセット制御部１３４はＳＣＰＵ２０の動作を停止するために信号ＡとＢを反転する。反転された信号Ａを受けてＳＣＰＵ２０のＲＯＭアドレス制御部２１４のアドレス更新動作が停止し、ＳＣＰＵ２０は停止する。また信号Ｂは“ＳＣＰＵ停止中”を示す信号としてＭＣＰＵ１０のオペレーション制御回路１１２に与えられる。ＭＣＰＵ１０のインタラプト処理ルーチン（第５図）の５－３に示すＳＣＰＵ状態検査命令を実行する際、ＭＣＰＵ１０のオペレーション制御回路１１２はＳＣＰＵ状態フラグＢを読む。フラグＢが“ＳＣＰＵ停止中”を示し、したがって、ＳＣＰＵ２０での音源処理（第６図）が完了しているときにＭＣＰＵ１０は５－４に進んでＳＣＰＵ２０の生成した楽音波形データを読み込む。ＭＣＰＵ１０は第５図のインタラプト処理ルーチン終了時にオペレーション制御回路１１２からＲＯＭアドレス制御部１１４にメインプログラムへの復帰コマンド信号を与えて、中断していたメインプログラムに制御を戻す。

第８図に、時間の流れに沿う本実施例の動作の流れを示す。Ａ～Ｆはメインプログラムの断片である。５Ａ～５Ｆは第５図のＭＣＰＵインタラプト処理ルーチンを表わし、６Ａ～６Ｆは第６図のＳＣＰＵインタラプト処理ルーチンを表わす。図示のように、インタラプト信号ＩＮＴが発生すると、ＭＣＰＵ１０は実行中のプログラムを中断し、インタラプト処理が各ＣＰＵ１０、２０において開始し、音源の並行処理が実行される。

第１２図に上述したＳＣＰＵの動作開始・終了機能を実現する構成を詳細に示し、第１３図～第１５図にその動作のタイムチャートを示す。第１３図のタイムチャートにおいて、ＣＫ１、ＣＫ２はＭＣＰＵ１０とＳＣＰＵ２０におけるクロック発生回路１３５、２３６に入力される２相のマスタークロックであり、このマスタークロックＣＫ１、ＣＫ２からクロック発生回路１３６はＭＣＰＵ１０動作のための基本タイミングを与える３相のクロックＴ１、Ｔ２、Ｔ３を生成する。この３相クロックの繰り返し周期がマシンサイクル（最短の命令実行時間）を定める。クロックＴ１ＣＫ１、Ｔ２ＣＫ２、Ｔ３ＣＫ３はそれぞれ、Ｔ１と

ＣＫ１、Ｔ２とＣＫ２、Ｔ３とＣＫ３の論理積信号である。オペレーションラッチ信号はＭＣＰＵ１０の制御用ＲＯＭ１０２のインストラクション出力ラッチ１０２ａにＲＯＭ１０２からのインストラクションをラッチさせるための信号である。第１３図には図示しないが、ＳＣＰＵ２０のクロック回路２３６も同様のクロック信号を生成する（第３図、第２５図参照）。なお、ＭＣＰＵ１０とＳＣＰＵ２０に共通のクロック発生回路を使用してもよい。

第１２図において、点線１６の右側はＳＣＰＵ２０であり左側はＭＣＰＵ１０である。左側の要素のうち、ラッチＬ１、ラッチＬ２、ゲート１１４２～１１５４はＭＣＰＵ１０（第２図）のＲＯＭアドレス制御部１１４に含まれる回路要素である。ラッチＬ１にはＭＣＰＵ１０の実行すべき次の命令のＲＯＭ１０２アドレス情報ＡＮ（ＲＯＭ１０２からの現命令に含まれる情報）がクロックＴ１ＣＫ１でラッチされる。メインプログラム（第４図）の実行中、ラッチＬ１の出力は次アドレスＢＮとしてＭＣＰＵ１０のＲＯＭアドレスデコーダ１０４に入力される。即ち、ラッチＬ１の出力はインバータ１１４４、３状態インバータゲート１１４６（イネーブルされている）を通ってＲＯＭアドレスデコーダ１０４へのアドレス入力ＢＮとなる。ここでインタラプト発生部１１６からインタラプト信号ＩＮＴが発生すると、この信号ＩＮＴを受けるＯＲゲート１１５４から、インバータ１１４８を介してラッチＬ１の出力側にある３状態インバータゲート１１４６をオフ（ハイインピーダンス）にする信号が加えられ、代りに、ＯＲゲート１１５４からの信号により、割込入口／戻先アドレス選択ゲート１１５０の出力側にある３状態インバータゲート１１５２がゲート１１５０の出力をＲＯＭアドレスデコーダ１０４のアドレス入力ＢＮに通す。割込入口／戻先アドレス選択ゲート１１５０はインタラプト信号ＩＮＴとラッチＬ２からの出力信号を受けるＮＯＲゲート群で構成され、“Ｈ”のインタラプト信号ＩＮＴ発生時に、インタラプト処理ルーチン（第５図）の入口（エントリポイント）を表わすオール“０”の信号を出力し、この信号は３状態インバータゲート１１５２で反転されて、オール“１”の信号ＢＮとしてＭＣＰＵのＲＯＭアドレスデコーダ１０４に入力される。そして次のオペレーションラッチ信号により、制御用ＲＯＭ１０２からインストラクション出力ラッチ１０２ａにインタラプト処理ルーチンの最初の命令がフエッチされる。以上により、ＭＣＰＵ１０の制御がインタラプト処理ルーチンに移る。

更に、インタラプト発生部１１６からのインタラプト信号ＩＮＴはクロックＴ２ＣＫ２のタイミングでＡＮＤゲート１１４２を通り、ラッチ信号としてラッチＬ２を動作させる。これにより、ラッチＬ２はバスＡＮ上にあるメインプログラムの次命令のアドレスをラッチ（退避）してメインプログラムを中断させる。

更にインタラプト発生部１１６からのインタラプト信号ＩＮＴはＳＣＰＵリセット制御部１３４に供給される。ＳＣＰＵリセット制御部１３４は図示のように結合されたＤフリップフロップ１３４２、ＮＡＮＤゲート１３４４、Ｒ－Ｓフリップフロップ１３４６から成る。メインプログラムの実行中、Ｒ－Ｓフリップフロップ１３４６はリセット状態にある（Ｑ＝“Ｌ”）。なお、図示しないがＲ－Ｓフリップフロップ１３４６はシステムのパワーオン時にリセット状態に初期化される。インタラプト信号ＩＮＴは、クロックＴ２ＣＫ１のタイミングでＤフリップフロップ１３４２に取り込まれ、次のクロックＴ１ＣＫ１のタイミングでＮＡＮＤゲート１３４４から反転されて出力され、Ｒ－Ｓフリップフロップ１３４６をセットする。この結果、Ｒ－Ｓフリップフロップ１３４６の$\overline{Q}$出力、即ち信号Ａが“Ｈ”から“Ｌ”に切り換え、Ｑ出力、即ちＳＣＰＵ状態フラグが“Ｌ”（ＳＣＰＵ停止中を示す）から“Ｈ”（ＳＣＰＵ動作中を示す）に変化する。信号Ａは、ＳＣＰＵ２０における次命令のアドレスＳＡＮをラッチするためのラッチＬ３にリセット解除信号（ラッチＬ３のイネーブル信号）として入力され

る。この結果、ラッチＬ３は次のクロックＴ１ＣＫ１のタイミングでバスＳＡＮに乗っているＳＣＰＵプログラム（第６図）の最初の命令のアドレスをＳＢＮとしてＳＣＰＵ２０のＲＯＭアドレスデコーダ２０４に入力する。このようにして、インタラプト発生部１１６からのインタラプト信号ＩＮＴに応答してＳＣＰＵ２０の動作が開始し、第５図に示す音源処理が実行される。

ＳＣＰＵ２０が音源処理の最後の命令を実行する際、ＳＣＰＵ２０のオペレーション制御回路１１２の内部で動作終了信号（復帰コマンド信号）ＳＲＴが発生する。この信号ＳＲＴはＤフリップフロップ２１２２にクロックＴ２ＣＫ１のタイミングで取り込まれた後、次のＴ１ＣＫ１のタイミング（次のダミー命令のラッチタイミング）で動作するＮＡＮＤゲート２１２４で反転され、ローパルスの動作終了信号ＢとしてＳＣＰＵリセット制御部１３４のＲ－Ｓフリップフロップ１３４６をリセットする。この結果、Ｒ－Ｓフリップフロップ１３４６の$\overline{Q}$出力、即ち、信号Ａは“Ｌ”から“Ｈ”に切り換り、Ｑ出力、即ち、ＳＣＰＵ状態フラグはＳＣＰＵ動作中を示す“Ｈ”からＳＣＰＵ２０停止中を示す“Ｌ”に切り換る。“Ｈ”レベルの信号Ａ（リセット信号）により、ラッチＬ３の動作は禁止され、ラッチＬ３出力、つまり、アドレスデコーダ２０４の入力はダミー命令の（ＮＯＰ命令）のアドレスに固定される。このときラッチＬ３の入力バスＳＡＮにはＳＣＰＵ音源処理プログラム（第６図）の最初の命令のアドレス情報（ＮＯＰ命令語に含まれる）が乗っている。

ＭＣＰＵ１０はインタラプト処理ルーチン（第５図）のＳＣＰＵ状態検査命令５－３の実行時にオペレーション制御回路１１２を介してＳＣＰＵ状態フラグのレベルを検査し、ＳＣＰＵの停止中、即ちＳＣＰＵ２０の音源処理の完了を確認してから、ＳＣＰＵ２０の処理結果である楽音波形データをＲＡＭ２０６からＲＡＭ１０６に読み取る（５－４）。これによりＭＣＰＵ１０はＳＣＰＵ２０の正しい処理結果を効率よく得ることができる。

ＭＣＰＵ１０はインタラプト処理ルーチンの最後の命令の実行時に、オペレーション制御回路１１２から復帰コマンド信号ＲＴのパルスを発生する。この信号パルスＲＴはＯＲゲート１６５４、インバータ１１４８を通って、ラッチＬ１の出力側のアドレスゲート１１４６を一時的にオフし、代りに、ラッチＬ２に結合する割込入口／戻先アドレス選択ゲート１１５０の出力側にあるアドレスゲート１１５２を一時的に開く。この時点で、割込入口／戻先アドレス選択ゲート１１５０はラッチＬ２に退避してあった中断されたメインプログラムの命令のアドレスを反転して通すインバータとして働き、このゲート１１５０の反転出力が信号パルスＲＴによりインバータとして働く３状態ゲート１１５２において再度反転される。この結果、ＭＣＰＵ１０のＲＯＭアドレスデコーダ１０４には中断されていたメインプログラムの命令のアドレスが入力され、次のオペレーションラッチ信号により、制御用ＲＯＭ１０２からインストラクション出力ラッチ１０２ａを介してその命令が取り出される。このようにして、ＭＣＰＵ１０の制御はメインプログラムに復帰する。

以上のように、本実施例の電子楽器処理装置は、ＭＣＰＵ１０によるＳＣＰＵ２０の動作期間の管理をＳＣＰＵリセット制御部１３４のような簡単な管理インターフェース構成を設けることで効率よく、確実に行うことができる。

＜複数データ転送＞

ＣＰＵを用いたある種のアプリケーションでは、ＣＰＵはメインプログラム（第１のプログラム）の実行において複数のデータを更新し、インタラプト処理ルーチン（第２のプログラム）の実行において、その処理の目的のためにこれら複数のデータを参照する。これはメインプログラムからインタラプト処理ルーチンへデータを渡す問題である。このような複数のデータは、インタラプト処理ルーチンによってメインプログラムが中断される前に、メインプログラムにおいて全てのデ

ータを更新しなければならない。メインプログラムが複数のデータの一部だけを更新した時点で中断されてインタラプト処理ルーチンにＣＰＵの制御が移ってしまうとインタラプト処理ルーチンの処理結果は誤ったものになる。

本実施例の電子楽器処理装置の場合も、ＭＣＰＵ１０のメインプログラム（第４図）からＭＣＰＵ.１０のタイマインタラプト処理ルーチン（第５図）（及び第６図に示すＳＣＰＵ２０のタイマインタラプト処理ルーチン）に渡す複数のデータがある。エンベロープ△ｘ（エンベロープ演算周期）、加減フラグ付エンベロープ△ｙ（エンベロープ変化分）目標エンベロープから成るエンベロープパラメータはその例である。データ源である外部データメモリ９０はエンベロープのセグメント（例えばアタックセグメント、ディケィセグメント、サスティンセグメント等）ごとにエンベロープパラメータを記憶している。ＭＣＰＵ１０のメインプログラムは発音処理４－９において、押鍵（ノートオン）あるいはインタラプト処理ルーチンのチャンネル音源処理（第９図）内で検出されたエンベロープの目標値への到達（９－６，９－７参照）に応答して所定のセグメントについてのエンベロープパラメータ（新しい目標エンベロープ、エンベロープ△ｘ，加減フラグ付エンベロープ△ｙ）を外部データメモリ９０から取り出してＭＣＰＵ内部ＲＡＭ１０６（またはＳＣＰＵ内部ＲＡＭ２０６）の対応するチャンネル音源処理レジスタにセットすることによって複数のデータから成るエンベロープパラメータを更新する必要がある。このような複数のデータはインタラプト発生部１１６からのインタラプト信号ＩＮＴによってメインプログラムが中断される前に、メインプログラムにおいて更新を完了させておかなければならない。

このような複数のデータ転送（更新）の問題を解決するために、本実施例では２つの解決手段を開示する。第１の解決手段はデータ更新の間、インタラプトをマスクしてメインプログラムのデータ更新命令群の実行が中断されないようにするインタラプトマスク方式であり、第２の解決手段は複数のデータ転送を一命令で実行する機能を利用した一命令方式である。

<u>インタラプトマスク方式（第１６、第１７、第２～第７図）</u>

この方式によれば、インタラプト発生部１１６からのインタラプトはメインプログラム、特に発音処理４－９におけるデータ更新命令群によって内部ＲＡＭのチャンネル音源レジスタ群にデータをセットする間、マスクされて、ＭＣＰＵ１０の制御がメインプログラム（第４図）からインタラプト処理ルーチン（第５図）に移るのが禁止される。

第１７図に複数のデータ転送を含むエンベロープ処理（メインプログラムの発音処理４－９内にある）のフローを示し、第１６図にインタラプトマスクに関連するハードウェアを示す。

第１７図においてＭＣＰＵ１０は１７－１で指定音源チャンネルの現在エンベロープが目標エンベロープに到達しているかどうかを調べる。到達すればＭＣＰＵ１０は１７－２に進み、外部データメモリ９０（第１図）から、次のエンベロープセグメントに関するエンベロープパラメータ、即ち、新しい目標エンベロープ、加減フラグ付エンベロープ△ｙ、エンベロープ△ｘを取り出し、内部ＲＡＭ１０６内の転送バッファにセットする。ここに転送バッファはデータ源とデータ目的地との間の中間的な記憶部でありインタラプト処理ルーチン（第９図）によって参照されないＲＡＭ領域であるので、この時点でのインタラプトマスクは不要である。転送バッファを設けた理由はデータ源であるメモリ９０がＭＣＰＵ１０とＳＣＰＵ２０によって共用される外部メモリであり、そのデータアクセス時間が内部ＲＡＭ相互のデータ転送時間より長くなること等による。ブロック１７－２の機能は外部データメモリ９０から内部ＲＡＭ１００への複数のデータ転送命令を順次実行することで処理される。

転送バッファからチャンネル音源用レジスタ群

（インタラプト処理ルーチンにおいて参照される）へのデータ転送はブロック１７－４で実行される。このデータ転送中にＭＣＰＵ１０の制御がタイマインタラプト処理ルーチン（第５図）に移行しないようにするため（あるいはＳＣＰＵ２０の制御が第６図のプログラムに移行しないようにするため）、ＭＣＰＵ１０はブロック１７－４に先立ってブロック１７－３でインタラプトをマスクする命令を実行する。このインタラプトマスク命令の実行中に、ＭＣＰＵ１０のオペレーション制御回路１１２からローアクティブのマスク信号ＭＡＳＫが発生する。このマスク信号ＭＡＳＫはインタラプト発生部１１６からのインタラプト信号ＩＮＴをマスクして、インタラプト処理ルーチン（第５図、第６図）への制御の移行を禁止するように作用する。この目的のため、第１６図において、インタラプト発生部１１６に結合するマスク解除待機部１５０が設けられる。マスク解除待機部１５０は図示のように結合したＲ－Ｓフリップフロップ１５０２、ＡＮＤゲート１５０４、及びＤフリップフロップ１５０６を含む。

マスク信号ＭＡＳＫがマスク解除を示す“Ｈ”レベルのとき、インタラプト発生部１１６からのインタラプト信号ＩＮＴにより、Ｒ－Ｓフリップフロップ１５０２がセットされ、その出力が“Ｈ”のＭＡＳＫによりイネーブルされているＡＮＤゲートを通って、Ｄフリップフロップ１５０６にＴＩＣＫ１のタイミングで取り込まれ、このＤフリップフロップ１５０６の出力が、実際のインタラプト信号Ａ－ＩＮＴとしてＭＣＰＵ１０のＲＯＭアドレス制御部１１４に入力される。その結果、ＳＣＰＵ動作開始・終了機能のところで述べたように、ＲＯＭアドレス制御部１１４のゲート１１５２からＲＯＭアドレスデコーダ１０４にインタラプト処理ルーチン（第５図）のエントリポイントのアドレスが入力されるとともに、次のメインプログラム命令のアドレスがバスＡＮからラッチＬ２に退避されて、ＭＣＰＵ１０の制御がインタラプト処理ルーチンに移行し、メインプログラムは中断される。また、信号Ａ－ＩＮＴはＳＣＰＵリセット制御部１３４に入力され、その結果、ＳＣＰＵ動作開始・終了機能のところで述べたようにＳＣＰＵ２０のプログラム（第７図）動作が開始する。Ｄフリップフロップ１５０６からのＨレベルの出力はＲ－Ｓフリップフロップ１５０２をリセットし、その結果、次のＴＩＣＫ１のタイミングでＤフリップフロップ１５０６の出力（マスク解除待機部１５０の出力）はＬレベルに切り換る。

これに対し、第１７図の１７－３に示すようにインタラプトマスク命令の実行により、オペレーション制御回路１１２からローアクティブのマスク信号ＭＡＳＫがマスク解除待機部１５０に入力される場合には、インタラプト発生部１１６からのインタラプト信号はＡＮＤゲート１５０４によってマスクされる。その結果、マスク解除待機部１５０４はマスク信号ＭＡＳＫがローアクティブの間、その出力Ａ－ＩＮＴを“Ｌ”の割込禁止レベルにし、ＲＯＭアドレス制御回路１１４の通常動作を継続させ、ＭＣＰＵ１０に対するメインプログラムの制御を続行させる。

したがって、ブロック１７－４に示す転送命令群（及びエンベロープ△ｘ用タイマーのクリア命令）の実行は、実行の途中で、インタラプト発生部１１６からインタラプト信号ＩＮＴが発生した場合にも中断されない。これにより、インタラプト処理ルーチン（第５図、第６図）は正しく更新されたエンベロープパラメータを参照でき、正しい演算結果（楽音波形データ）を得ることができる。

しかる後、ＭＣＰＵ１０はブロック１７－５に示すインタラプトマスク解除命令を実行する。この結果、オペレーション制御回路１１２からマスク解除待機部１５０に供給される信号ＭＡＳＫはマスク解除を示す“Ｈ”レベルに切り換る。複数のデータ転送を含むブロック１７－４の実行中に、インタラプト発生部１１６からインタラプト信号が発生したような場合には、マスク解除待機部１５０のＲ－Ｓフリップフロップ１５０２の出力によって、このマスク解除命令の実行後にイン

タラプトの要求が受け付けられ、上述したようにしてメインプログラムが中断され、インタラプト処理ルーチンに制御が移行する。

一命令方式（第１８～第２１図）

この方式はメインプログラム（第４図）において複数のデータをインタラプト処理ルーチンの参照する内部ＲＡＭ領域にセットするために、ロング命令と呼ばれる複数データー括転送のための単一命令を利用し、ロング命令の実行が終了するまでインタラプト処理ルーチンにＭＣＰＵ１０の制御が移行しないようにしたものである。

単一の命令（ロング命令）で複数のデータ転送が可能なＣＰＵは例えば特公昭６０－４７６１２号に開示されており、本実施例にこの技術が適用できる。特公昭６０－４７６１２号によれば、ロング命令は連続するアドレスにある複数のレジスタ間（例えばレジスタＡ０～Ａ３をレジスタＢ０～Ｂ３）の転送に適用可能である（ここにレジスタとはＲＡＭの１記憶場所を意味し、Ａ、ＢはＲＡＭのアドレス上位、即ち行アドレスを表わし、０、３はＲＡＭのアドレス下位、即ち列アドレスを表わす）。制御用ＲＯＭ（本実施例の要素１０２に対応する）からのロング命令語にはソースレジスタの行アドレス（上の例でいえばＡ）、ディスティネーションレジスタの行アドレス（Ｂ）、最初のデータ転送に係るレジスタの列アドレス（０）、最後のデータ転送に係るレジスタの列アドレス（３）の情報が含まれる。ＲＡＭアドレス制御部（本実施例の要素１０５に対応する）はロング命令の実行に適するように構成され、列アドレスを最初の転送の列アドレスから最後の転送の列アドレスまでデータ転送の都度、１ずつ更新するカウンタ（その出力がＲＡＭの列アドレス入力に順次加えられる）と、すべてのデータ転送が完了したことを検出するためカウンタ出力と最後のデータ転送の列アドレス値とを比較し、一致したときにロング命令実行完了信号を発生する一致回路とを含んでいる。

以下の説明において、本実施例の制御用ＲＯＭ１０２のメインプログラム内には上述したようなロング命令が含まれるものとし、ＲＡＭアドレス制御部１０５、２０５は上述したようにロング命令の実行を適用できるように構成されているものとする。

第１８図にロング命令の実行中、インタラプト信号ＩＮＴによるメインプログラムの中断を禁止する回路を含むハードウェアのブロック図を示し、第１９図にロング命令をエンベロープパラメータの転送に適用した場合のＲＡＭのメモリマップを示し、第２０図にロング命令（単一転送命令）と複数の転送命令との動作の比較を示し、第２１図にロング命令を使用したエンベロープパラメータの転送に関連するフローチャートを示す。

第１８図において、インタラプト発生部１１６に転送終了待機部１５２が結合している。この回路１５２はロング命令の実行中、インタラプト信号によるメインプログラムの中断を禁止する。転送終了待機部１５２は図示のようにＲ－Ｓフリップフロップ１５２２、ＡＮＤゲート１５２４、Ｄフリップフロップ１５２６から成り、Ｄフリップフロップ１５２６の出力（転送終了待機部１５２の出力）が実際に作用するインタラプト信号Ａ－ＩＮＴとしてアドレス制御部２１４とＳＣＰＵリセット制御部１３４に結合している。ＡＮＤゲート１５２４に入力される信号～ＬＯＮＧが“Ｌ”の間は、インタラプト発生部１１６からインタラプト信号ＩＮＴが発生しても、Ｄフリップフロップ１５２６の出力は“Ｌ”のままであり、ＲＯＭアドレス制御部２１４とＳＣＰＵリセット制御部１３４はインタラプト信号ＩＮＴの作用を受けない。ここに、信号～ＬＯＮＧはロング命令の実行中に“Ｌ”となる信号であり、ロング命令の実行完了に伴ってＲＡＭアドレス制御部１０４の一致回路から発生するロング命令実行完了信号に応答して“Ｈ”に復帰する。信号～ＬＯＮＧのレベルが“Ｈ”のときには、インタラプト発生部１１６からのインタラプト信号ＩＮＴは転送終了待機部１５２を通ってＲＯＭアドレス

制御部２１４とＳＣＰＵリセット制御部１３４に作用し、ＭＣＰＵ１０の制御をメインプログラム（第４図）からインタラプト処理ルーチン（第５図）に移行させ、ＳＣＰＵ２０のプログラム（第６図）動作を開始させる。

エンベロープパラメータの更新に一命令方式を適用する場合において、インタラプト処理ルーチン（第５図、第６図）のチャンネル音源処理サブルーチン（第９図）が参照し、メインプログラムのエンベロープ処理サブルーチン（第２１図）が設定（更新）するエンベロープパラメータはエンベロープムｘ用タイマー、新目標エンベロープ、新エンベロープムｘ、新加減フラグ付エンベロープムｙである。本実施例において、これらのエンベロープパラメータのデータ源は外部メモリ９０（第１図）にある。エンベロープパラメータの更新の際に（２１－１）。外部データメモリ９０から内部ＲＡＭ１０６、２０６のチャンネル音源データ領域への直接の転送は望ましくないので、外部データメモリ９０からのエンベロープパラメータはいったん内部ＲＡＭ１０６内の転送用バッファ領域に移し（２１－２）、次に、転送用バッファ領域からチャンネル音源データ領域に移す（２１－３）。

この転送用バッファ領域からチャンネル音源データ領域へのデータ転送処理２１－３に上述したロング命令が使用される。ロング命令を適用するために、転送用バッファ領域はＲＡＭ上の連続した領域であることを必要とし、同様にエンベロープパラメータのチャンネル音源データ領域も連続した領域であることを必要とする。この例を第１９図に示す。ここでは、エンベロープパラメータの転送用バッファ領域は、レジスタＸ４～Ｘ７の連続領域にマッピングされエンベロープパラメータについての１チャンネル音源データ領域はレジスタＡ４～Ａ７の連続領域にマッピングされている。したがって、１チャンネルでエンベロープパラメータを更新する必要のあるときには、２１－３で、レジスタＸ４～Ｘ７をレジスタＡ４～Ａ７に転送するロング命令を実行すればよい。この命令が実行されている間は、上述したようにインタラプト信号ＩＮＴがインタラプト発生部１１６から発生しても、転送終了待機部１５２のロング命令完了待機機能により、ロング命令が終了するまではインタラプト信号の作用がＲＯＭアドレス制御部１１４、ＳＣＰＵリセット制御部１３４に波及しない（第２０図（Ｂ）参照）。この結果、チャンネル音源データ領域のエンベロープパラメータが全て正しい更新値に変更された後にインタラプト処理ルーチンが開始するので、その演算結果（楽音波形データ）が正しい値を示し、誤りのない動作が保証される。

これに対しもし、２１－３に示す転送処理機能を複数の転送命令（一命令ごとに１つのエンベロープパラメータを転送する）の実行によって果たそうとした場合には、転送の途中で、例えば、第２０図（Ａ）に示すように転送命令１の実行中にインタラプト信号ＩＮＴが発生すると次のマシンサイクルで転送命令２の代りにインタラプト処理ルーチンの最初の命令が実行されてエンベロープ転送処理は途中で中断されてしまう。この結果、インタラプト処理ルーチンの処理結果（楽音波形データ）は誤った値となってしまう。

一命令方式による複数データの転送（更新）処理では１７－３、１７－５に示すようなインタラプトマスク命令、インタラプト解除命令を実行する必要がなく、オーバーヘッドなしの最短時間で、転送処理を実行することができる利点もある。

変形例として、第１８図に示すような転送終了待機部１５２の代りに、ロング命令の実行中、制御用ＲＯＭ１０２、２０２からの命令をフェッチするインストラクション出力ラッチ１０２ａの動作を禁止する手段を使用してもよい。即ち、制御用ＲＯＭ１０２からラッチ１０２ａを介して与えられるロング命令語に含まれるモード信号（命令がロングであることを示している）によって、インストラクション出力ラッチ１０２ａ、２０２ａに加えるオペレーションラッチ信号の発生を禁止し、ロング命令の実行完了信号に応答して次のマ

シンサイクルでオペレーションラッチ信号を発生する回路をオペレーション制御回路１１２内に設ければ、インタラプト信号ＩＮＴがロング命令の実行中に発生しても制御用ＲＯＭ１０２、２０２からインタラプト処理ルーチンの最初の命令語はロング命令の実行が終了するまではインストラクション出力ラッチ１０２ａ、２０２ａにフェッチされない（したがって実行もされない）ので実施例と同様の効果が得られる。

＜ＭＣＰＵからのＳＣＰＵアクセス機能＞

本実施例の装置はＭＣＰＵ１０からＳＣＰＵ２０の内部ＲＡＭ２０６にデータを高速にアクセス（リードまたはライト）する機能を有している。この課題は一般に複数のＣＰＵ間のデータアクセス問題として把えられている。従来技術ではこの種のインターＣＰＵデータアクセスに時間がかかる問題がある。従来技術ではアクセスを要求するＣＰＵからアクセスを要求されるＣＰＵに対し、要求信号を与える。アクセスを要求されるＣＰＵはこの要求信号に対し、ただちに要求側ＣＰＵからのデータアクセスを許可する承認信号を発生することはできず、実行中のオペレーションが完了するまで承認信号の発生を遅延させる。したがって、従来のインターＣＰＵデータアクセス方式は高速処理が要求されるアプリケーションにおける障害の１つとなっている。

本実施例では高速のインターＣＰＵデータアクセスのために２つの解決手段、即ち、ＳＣＰＵ停止モード利用方式と瞬時強制アクセス方式を開示する。

ＳＣＰＵ停止モード利用方式（第２２図、第２、第３図）

この方式は上述したＳＣＰＵ動作開始・終了機能を利用したものである。この機能によりＳＣＰＵ２０のプログラム（第６図）動作はＭＣＰＵ１０におけるインタラプト処理ルーチン（第５図）の開始と同時に開始し、ＭＣＰＵ１０のインタラプト処理ルーチンが終了する前に終了する。したがって、ＭＣＰＵ１０においてメインプログラム（第４図）が動作している間はＳＣＰＵ２０は停止モード（リセット状態）にある。第２図に示すように停止モード中では、リセット制御部１３４からの信号Ａが“ＳＣＰＵ停止中”を示す“Ｈ”レベルになる。この信号Ａにより、ＳＣＰＵ２０（第３図）ではＲＯＭアドレス制御部２１４の動作が停止し、ＲＡＭアドレス制御部２０４はＳＣＰＵ２０の制御用ＲＯＭ２０２からのＲＡＭアドレスバスＳＡではなく、ＭＣＰＵ１０からバスゲート１２８を介してＲＡＭアドレスバスＭａに結合してＭＣＰＵ１０からのＳＣＰＵ内部ＲＡＭ２０６の指定アドレスを受けるように動作モードが設定され、ＲＡＭデータに切り換え部２４０はＳＣＰＵ２０のオペレーション結果（ＡＬＵ部２０８出力または乗算器２１０出力）を運ぶデータバスＤＢではなくＭＣＰＵ１０からのデータを運ぶデータバスＤ$_{out}$にＲＡＭ２０６のデータインを結合する動作モードに設定され、ライト信号切り換え部２４２はＳＣＰＵオペレーション制御回路２１２からのリード／ライト制御信号ではなくオペレーション制御回路１１２からのリード／ライト制御信号ＣをＲＡＭ２０６のリード／ライト制御入力に結合する動作モードに設定される。このように停止状態のとき、ＳＣＰＵ２０はＭＣＰＵ１０によってデータアクセスが可能な状態に置かれている。

したがって、本実施例によれば、ＭＣＰＵ１０はメインプログラムにおいてＳＣＰＵ２０の内部ＲＡＭ２０６を自由にアクセスすることができる。この様子を第２２図に示す。ＳＣＰＵ２０の停止状態（音源処理完了）の確認、即ちＭＣＰＵオペレーション制御回路１１２におけるＳＣＰＵリセット制御部１３４からのＳＣＰＵ状態フラグの検査はＭＣＰＵ１０のインタラプト処理ルーチン（第５図）のなかで１回だけ行えばよい（５－３参照）。いったん停止状態が確認されれば、次のインタラプト信号ＩＮＴが発生するまで、再度の確認をする必要なしに、一命令の実行で、ＭＣＰＵ１０はＳＣＰＵ２０の内部ＲＡＭ２０６をア

クセスできる。したがって、従来に比べ、ＳＣＰＵ２０へのデータアクセスに要する時間が大幅に短縮される。

瞬時強制アクセス方式（第２３～第２５図）

この方式はデータアクセスのためにＭＣＰＵ１０とＳＣＰＵ２０との間で従来のようなアクセスの要求と承認という手続を踏むことなく、ＭＣＰＵ１０からのＳＣＰＵデータアクセス時にＳＣＰＵ２０の動作を強制的に一時停止させ、その間にＭＣＰＵ１０がＳＣＰＵ２０の内部ＲＡＭ２０５にアクセスするものである。この方式によれば、ＭＣＰＵ１０は任意のときにＳＣＰＵ２０の状態を調べる必要なしにＳＣＰＵ２０を高速に（一命令実行で）アクセスできる。

このような特徴を備えたＭＣＰＵ１０のブロック図とＳＣＰＵ２０のブロック図をそれぞれ第２３図と第２４図に示す。なお、このＭＣＰＵとＳＣＰＵは上述したＳＣＰＵ動作開始終了機能に関する要素（第２図のＳＣＰＵリセット制御回路１３４その他）を含むが第２３図と第２４図では簡略化のため図示を省略してある。この場合、リセット制御回路１３４からのＳＣＰＵ動作起動／停止信号ＡはＳＣＰＵ２０（第２４図）のＲＯＭアドレス制御部２１４にのみ供給すれば十分である。第２３図と第２４図のＭＣＰＵ１０とＳＣＰＵ２０の瞬時強制アクセスに関する動作のタイムチャートを第２５図に示す。

瞬時強制アクセス方式を使用する場合、ＭＣＰＵ１０とＳＣＰＵ２０は別個のクロック発生回路１３６、２３６Ｍを必要とする。ＳＣＰＵ２０のクロック発生回路２３６Ｍは、ＳＣＰＵ２０へのデータアクセス命令実行時にＭＣＰＵ１０のオペレーション制御回路１１２Ｍから出力されるハイアクティブのＳＣＰＵアクセス信号Ｄに応答してその動作を停止する。これに関連し、ＭＣＰＵ１０のクロック発生回路１３６とＳＣＰＵ２０のクロック発生回路２３６Ｍは共通の２相マスタークロック信号ＣＫ１、ＣＫ２を受けるが、出力するクロックのタイミングは独立である。ＭＣＰＵ１０ではクロック発生回路１３６からの３相のクロック信号Ｔ１、Ｔ２、Ｔ３の一周期でマシンサイクル（最短の一命令実行時間）が規定され、一方、ＳＣＰＵ２０ではクロック発生回路２３６Ｍからの３相のクロック信号ＳＴ１、ＳＴ２、ＳＴ３の一周期でそのマシンサイクルが規定される。第２５図において、ＳＣＰＵアクセス信号Ｄが発生する前において、ＭＣＰＵ１０に関するクロックＴ１のタイミングはＳＣＰＵ２０に関するクロックＳＴ１ではなくクロックＳＴ２のタイミングに一致している。両ＣＰＵ間で取り得る他のタイミング関係はＴ１がＳＴ１に一致する関係とＴ１がＳＴ３に一致する関係である。

ＭＣＰＵ１０におけるＳＣＰＵアクセス命令実行中にオペレーション制御回路１１２から出力されるＳＣＰＵアクセス信号Ｄは、ＳＣＰＵ２０のクロック発生回路２３６Ｍを停止させてＳＣＰＵ２０で実行中のオペレーションを停止させるとともに、その停止中にＭＣＰＵ１０がＳＣＰＵ２０の内部ＲＡＭ２０６をアクセスできるように、ＭＣＰＵ１０からの内部ＲＡＭ２０６の指定アドレスに係るパスゲート１２８、ＳＣＰＵ内部ＲＡＭ２０６に対するアドレス制御部２０４、データイン切り換え部２４０、及びライト信号切り換え部２４２の各動作モードを“ＳＣＰＵ側”から“ＭＣＰＵ側”に切り換える機能を有する。このために、ＳＣＰＵアクセス信号はこれらの要素１２８、２０４、２４０、２４２の動作モードを選択する制御入力にＤフリップフロップ２５０とＡＮＤゲート２５２とから成る遅延回路を介して結合している。このようなアクセス可能状態の下で、ＭＣＰＵ１０はパスゲート１２８、ＲＡＭアドレス制御部２０４を介してＳＣＰＵ内部ＲＡＭ２０６をアドレッシングし、リードアクセスの場合にはＳＣＰＵ内部ＲＡＭ２０６から出力されるデータをパスゲート１３２を介してＭＣＰＵ内部ＲＡＭ１０６に読み込み、ライトアクセスの場合には、パスゲート１３０を介して書き込みデータをデータバスD_{out}に乗せ、ＳＣＰＵ内部ＲＡＭ２０５にライト信号Ｃを与えてデータを書き込

む。

ＭＣＰＵ１０からのＳＣＰＵアクセス信号Ｄによって ＳＣＰＵ２０のオペレーションを中断する場合に、オペレーションの中間結果が失われないようにする必要があり、ＳＣＰＵアクセス信号Ｄの解除後に、予め保持した中間結果を用いてＳＣＰＵ２０がオペレーションの残りの部分を実行できるようにする必要がある。このために、ＳＣＰＵ内部ＲＡＭ２０６のデータ出力を一時的に記憶するラッチ２０６ａ、２０６ｂを設けている。ラッチ２０６ａはＲＡＭ２０６からの演算数（第１オペランド）をＳＴ１ＣＫ１のタイミングでラッチし、ラッチ２０６ｂはＲＡＭ２０６からの被演算数（第２オペランド）をＳＴ２ＣＫ１のタイミングでラッチする。

第２５図を参照して動作例を述べると、この例では、ＭＣＰＵ１０はＳＣＰＵアクセス信号Ｄがハイアクティブレベルの間にＳＣＰＵ２０の内部ＲＡＭ２０６に対するライトアクセスを実行している。ＭＣＰＵ１０ではこのデータ書込オペレーションの最初のタイムスロットＴ１の間に、ＭＣＰＵ内部ＲＡＭ１０６から転送データ（ＲＡＭ２０６に書き込むべきデータ）を取り出す。次のタイムスロットＴ２でＭＣＰＵ１０はＳＣＰＵ内部ＲＡＭ２０６をアドレッシングする。最後のタイムスロットＴ３でＭＣＰＵ１０はＳＣＰＵ内部ＲＡＭ２０６にライト信号Ｃを与えてＲＡＭ２０６にデータを書き込む。ＳＣＰＵ２０側にとってＭＣＰＵ１０からのＳＣＰＵアクセス信号ＤはＳＣＰＵ２０のオペレーション２がタイムスロットＳＴ２に移るときはアクティブに変化している。このオペレーション２はＳＣＰＵ２０のＲＡＭ２０６にある被演算数と演算数をＡＬＵ部２０８または乗算器２１０で演算するような命令のオペレーションであり得る。ＭＣＰＵ１０からのＳＣＰＵアクセスタイムの直前のタイムスロットであるオペレーション２の最初のタイムスロットＳＴ１でＳＣＰＵ２０はＲＡＭ１０６から演算数のデータを取り出し、そのデータをクロックＴ１ＣＫ１により演算数ラッチ１０６ａにラッチしている。ＭＣＰＵ１０からのＳＣＰＵアクセス信号Ｄが発生しなければ、ＳＣＰＵ２０は次のタイムスロットＳＴ２でＲＡＭ１０６から被演算数を取り出して被演算数ラッチ１０ｂにラッチし、最後のタイムスロットＳＴ３でＡＬＵ部１０８または乗算器１１０で演算を実行してＲＡＭ１０６の被演算数レジスタに書き込む。実際には図示のようにオペレーション２の最初のタイムスロットＳＴ１に続いてＭＣＰＵ１０からのＳＣＰＵアクセス信号Ｄが発生している。この場合、１つの対策はオペレーション２の残り２つのタイムスロットＳＴ２とＳＴ３で実行すべき処理をＳＣＰＵアクセス信号Ｄが除去されるまで、即ちＭＣＰＵ１０のＳＣＰＵアクセスオペレーションが終了するまで中断することである。この方式でもＭＣＰＵ１０はＳＣＰＵ２０をアクセスするオペレーションを最短時間（ＭＣＰＵ１０の内部ＲＡＭ１０６をアクセスするのと同じ時間）内に実行できるが、ＳＣＰＵ２０にとっては最適ではなくＭＣＰＵ１０からのＳＣＰＵアクセスオペレーションの都度、ＳＣＰＵ２０のオペレーションがタイムスロット３つ分遅延されることになる。都合のよいことに、ＭＣＰＵ１０のＳＣＰＵアクセスオペレーションの最初のタイムスロットＴ１で実行される処理はＳＣＰＵ２０に影響を与えない処理である。この特徴を利用し、実施例ではＭＣＰＵ１０からＳＣＰＵアクセス信号Ｄが与えられても、ＭＣＰＵ１０のタイムスロットＴ１の間は、ＳＣＰＵ２０自身のオペレーションが継続できるようにして、ＳＣＰＵ２０の動作遅れをできるだけ短かくしている。第２５図の例でいえば、ＳＣＰＵ２０はＭＣＰＵ１０のＳＣＰＵデータ書込オペレーションの最初のタイムスロットＴ１の間に、ＲＡＭ２０６から被演算数のデータを取り出し、ラッチ２０６ｂにクロックＳＴ２ＣＫ１を与えて被演算数をラッチさせている。その後、ＳＣＰＵクロック発生回路２３６の動作はＳＣＰＵアクセス信号Ｄが除去されるまで停止し、ＳＣＰＵ２０は待ち状態に置かれる。そしてこの待ち状態の間、ＳＣＰＵ２０の要素１２８、２６４、２４０、２４２はＳＣＰＵア

クセス信号Dにより"ＭＣＰＵ側"に切り換えられ、ＭＣＰＵ１０のＳＣＰＵデータ書込オペレーションにおけるタイムスロットＴ２、Ｔ３に関する処理が実行されてＳＣＰＵ内部ＲＡＭ２０６にＭＣＰＵ１０からのデータが書き込まれる。

ＭＣＰＵ１０からのＳＣＰＵアクセス信号Ｄが除去されると、ＳＣＰＵクロック発生回路２３６は動作を再開し、クロックＳＴ３を"Ｈ"に変化させる。更に、ＳＣＰＵアクセス信号Ｄの除去により、ＳＣＰＵ２０の要素１２８、２０４、２４０、２４２が"ＳＣＰＵ側"に戻され、ＳＣＰＵ２０自身の動作が可能な状態になる。そこでＳＣＰＵ２０はこのタイムスロットＳＴ３において、ＡＬＵ部２０８または乗算器２１０の演算出力をＲＡＭ２０６に書き込んでオペレーション２の残りの部分を実行する。

第２５図のタイムチャートに示すように、ＳＣＰＵ２０の動作がＭＣＰＵ１０からのＳＣＰＵアクセスオペレーションの都度、中断される時間はタイムスロット２つ分だけである。

なお、ＭＣＰＵ１０がＳＣＰＵ２０の内部ＲＡＭ２０６からデータを読み出すリードアクセスオペレーションの場合、そのタイムスロットＴ２でＭＣＰＵ１０はＳＣＰＵ内部ＲＡＭ２０６をアドレッシングし、タイムスロットＴ３でＭＣＰＵ内部ＲＡＭ１０６をアドレッシングしてＳＣＰＵ内部ＲＡＭ２０６からのデータをバスゲート１３２を介してＭＣＰＵ内部ＲＡＭ１０６に取り込む。

以上のように、瞬時強制アクセス方式によればＭＣＰＵ１０はＳＣＰＵ２０の内部ＲＡＭ２０６に対するアクセスをＭＣＰＵ自身のＲＡＭ１０６に対するアクセスと同様に最短時間内で実行でき、待ち時間命令を実行する必要がない。更に、瞬時強制アクセス方式によれば、ＳＣＰＵ２０のオペレーションを途中で中断し、ＭＣＰＵ１０のＳＣＰＵアクセスオペレーション後に、中断されたところからオペレーションを再開できる。したがって、ＭＣＰＵ１０はＳＣＰＵ２０に対するアクセスに先立ってＳＣＰＵ２０の状態を検査する必要はなく、任意のときに、例えば、インタラプト処理ルーチン（第５図）中でも自由にＳＣＰＵ２０をアクセスすることができる。

＜共用メモリアクセス競合解消機能（第２６、第２７図、第１図）＞

第１図において外部メモリ９０は複数のＣＰＵ、即ちＭＣＰＵ１０とＳＣＰＵ２０に共用されるデータメモリである。したがって外部データメモリ９０に対する複数のアクセス、即ち、ＭＣＰＵ１０からの外部データメモリ９０アクセスと、ＳＣＰＵ２０からの外部データメモリ９０アクセスをサポートする手段が必要である。更に、外部データメモリ９０を共用化する場合においてＭＣＰＵ１０とＳＣＰＵ２０とが外部データメモリ９０を同時にアクセスを試みることを許容するのが望まれる。ＭＣＰＵ１０とＳＣＰＵ２０との間で外部データメモリ９０に対する使用権（トークン）を交換する機能を設けることにより、ＭＣＰＵ１０とＳＣＰＵ２０が同時には外部データメモリ９０をアクセスしないようにすることもできるが、トークンの手続は外部データメモリアクセスのための準備時間を占めるので、外部データメモリアクセスに要するトータルの時間が長くなり、効率的でない。一方、ＭＣＰＵ１０とＳＣＰＵ２０による外部データメモリ９０の同時アクセスを許容する場合、メモリ９０自体は物理的に同時アクセス不能であるので、同時アクセスによるアクセス競合を解消する手段が必要となる。

これらの手段を実現するため、第１図に示すようにＭＣＰＵ１０からの外部メモリアドレス情報はアドレスバスＭＡ、ＭＣＰＵ外部メモリアドレスラッチ３０Ｍ、アドレス切り換え回路４０、アドレス変換回路６０を介して外部メモリ９０のアドレス入力に結合されており、外部メモリ９０からのデータ出力はデータ変換回路７０、ＭＣＰＵ外部メモリデータラッチ８０Ｍ、データバスＭＤを介してＭＣＰＵ１０に結合されている。一方、ＳＣＰＵ２０からの外部メモリアドレス情報はアドレスバスＳＡ、ＳＣＰＵ外部メモリアドレスラ

ッチ３０Ｓ、アドレス切り換え回路４０、アドレス変換回路６０を介して外部メモリ９０のアドレス入力に結合されており、外部メモリ９０からのデータ出力はデータ変換回路７０、ＳＣＰＵ外部メモリデータラッチ８０Ｓ、データバスＳＤを介してＳＣＰＵ２０に結合されている。そして、ＭＣＰＵ１０とＳＣＰＵ２０からの外部データメモリアクセス要求を表わす信号ＭＣＰＵ－ｒｏｍａとＳＣＰＵ－ｒｏｍａを受けるメモリ装置競合回避回路５０により、上記ＭＣＰＵ外部メモリアドレスラッチ３０Ｍは、ＳＣＰＵ外部メモリアドレスラッチ３０Ｓ、アドレス切り換え回路４０、ＭＣＰＵ外部メモリデータラッチ８０Ｍ、ＳＣＰＵ外部メモリデータラッチ８０Ｓが制御されるようになっている。このメモリ装置競合回避回路５０に上述したアクセスの競合を回避する機能が含まれている。

第２６図にメモリ装置競合回避回路５０のブロック図を示し、第２７図にアクセスの競合に対する動作のタイムチャートを示す。

第２６図において、メモリ装置競合回避回路５０には入力としてＭＣＰＵ１０からのアクセス要求信号ＭＣＰＵ－ｒｏｍａ、ＳＣＰＵ２０からのアクセス要求信号ＳＣＰＵ－ｒｏｍａ、更に、ＭＣＰＵリセット信号ＭＲＥＳ及びＳＣＰＵリセット信号ＳＲＥＳ（第１図において図示省略）が結合する。ＭＣＰＵリセット信号ＭＲＥＳはセットリセット回路（Ｒ－Ｓフリップフロップ）５０２とその出力に結合するセットリセット回路５０６をリセットし、信号ＭＣＰＵ－ｒｏｍａは、セットリセット回路５０２をセットする。セットリセット回路５０２はＭＣＰＵ１０からのアクセス要求を一時記憶し、出力側セットリセット回路５０６はセット状態において、ＭＣＰＵ１０からのアクセス要求が受け付けられて外部メモリデータアクセス制御信号発生回路５１０を介してアクセスのオペレーションが実行中であることを示す。同様にＳＣＰＵリセット信号ＳＲＥＳはセットリセット回路５０４とその出力に結合するセットリセット回路５０８をリセットし、信号ＳＣＰＵ－ｒｏｍａはセットリセット回路５０４をセットする。セットリセット回路５０４はＳＣＰＵ２０からのアクセス要求を一時記憶し、出力側セットリセット回路５０８はセット状態においてＳＣＰＵ２０からのアクセス要求が受け付けられアクセスのオペレーションが実行中であることを示す。

詳細に述べると、ＭＣＰＵアクセス要求セットリセット回路５０２のセット状態の出力"Ｈ"はＳＣＰＵアクセス実行セットリセット回路５０８がセット状態でないことを条件として、即ち、ＳＣＰＵ２０のアクセスオペレーションが実行中でないことを条件として（他入力が５０８からのインバータ５２２を介した反転入力に結合するＡＮＤゲート５２４を介して）ＭＣＰＵアクセス実行セットリセット回路５０６をＭＣＰＵアクセス実行状態にセットし、このＭＣＰＵアクセス実行セットリセット回路５０６をセットする信号により、ＯＲゲート５１２（他入力がリセット信号ＭＲＥＳに結合する）を介してＭＣＰＵアクセス要求セットリセット回路５０２をリセットする。同様に、ＳＣＰＵアクセス要求セットリセット回路５０４のセット状態の出力"Ｈ"はＭＣＰＵアクセス実行セットリセット回路５０６がセット状態でないことを条件として、即ちＭＣＰＵ１０のアクセスオペレーションが実行中でないことを条件として（他入力の１つが５０６からのインバータ５２０を介した反転入力に結合するＡＮＤゲート５２６）を介してＳＣＰＵアクセス実行セットリセット回路５０８をＳＣＰＵアクセス実行状態にセットし、このＳＣＰＵアクセス実行セットリセット回路５０８をセットする信号により、ＯＲゲート５１６（他入力がリセット信号ＳＲＥＳに結合する）を介してＳＣＰＵアクセス要求セットリセット回路５０４をリセットする。以上の構成により、片方のＣＰＵ（例えばＳＣＰＵ２０）からアクセス要求があっても、他方のＣＰＵ（ＭＣＰＵ１０）に関するアクセスオペレーションが実行中のときは、その実行が完了するまではアクセスを要求したＣＰＵ（ＳＣＰＵ２０）に関するアクセスオペレーションは実行されない。これによ

り、アクセスの競合が基本的に回避される。

更に、ＭＣＰＵ１０とＳＣＰＵ２０とが完全に同時にアクセスを要求する場合がある。このアクセス競合に対し、実施例では、ＭＣＰＵ１０からのアクセス要求を優先させ、ＭＣＰＵ１０のアクセスオペレーションを実行してから、ＳＣＰＵ２０のアクセスオペレーションを実行している。このために、ＭＣＰＵアクセス要求セットリセット回路５０２がセット状態のときはその出力信号“Ｈ”によりインバータ５２５を介してＡＮＤゲート５２６を禁止しており、セットリセット回路５０２がセット中のときはＳＣＰＵアクセス要求セットリセット回路５０４がセット状態でもＳＣＰＵアクセス実行セットリセット回路５０８がセットされないようにしている。

外部メモリデータアクセス制御信号発生回路５１０は、セットリセット回路５０６と５０８からの出力に結合し、いずれかのセットリセット回路の出力がセット状態“Ｈ”に変化すると、そのセット状態が示すＣＰＵアクセスのオペレーションを一連のシーケンスで実行する。外部メモリデータアクセス制御信号発生回路５１０から出力される信号ＣＥとＯＥは外部メモリ７からデータを出力するための制御信号であり、信号ＭＤＬはＭＣＰＵ外部メモリデータラッチ８０Ｍに外部メモリ９０からのデータをラッチするための制御信号であり、信号ＳＤＬはＳＣＰＵ外部メモリデータラッチ８０Ｓに外部メモリ９０からのデータをラッチするための制御信号である。外部メモリデータアクセス制御信号発生回路５１０はアクセスオペレーションの実行を終了するとＥＮＤ信号を発生する。このＥＮＤ信号により、セット状態にあったアクセス実行セットリセット回路はリセットされる。即ち、ＥＮＤ信号は他入力がセットリセット回路５０６の出力に結合するＡＮＤゲート５２８と他入力がＭＣＰＵリセット信号ＭＲＥＳに結合するＯＲゲート５１４を介してセットリセット回路５０６のリセット入力に結合し、また他入力がセットリセット回路５０８の出力に結合するＡＮＤゲート５３０と他入力がＳＣＰＵリセット信号ＳＲＥＳに結合するＯＲゲート５１８を介してセットリセット回路５０８のリセット入力に結合する。

ＳＣＰＵアクセス実行セットリセット回路５０８の出力はインバータ５３２を介してアドレス切り換え回路４０に対するアドレス選択信号ＭＳＥＬを発生する。したがって、アドレス切り換え回路４０は、ＳＣＰＵ２０のアクセスオペレーションが実行中のときに、ＳＣＰＵ外部メモリアクセス用アドレスラッチ３０５からのＳＣＰＵアドレスを選択し、その他の場合はＭＣＰＵ外部メモリアクセス用アドレスラッチ３０Ｍからのアドレスを選択する。

第２７図の場合、ＭＣＰＵ１０とＳＣＰＵ２０は“ＭＣＰＵオペレーションのｒｏｍａ”、“ＳＣＰＵオペレーションのｒｏｍａ”に示すように同時に外部メモリ９０に対するアクセスを要求している。このｒｏｍａ命令のオペレーションにおいて、ＭＣＰＵ１０はアドレスバスＭＡにアドレス情報を送出し、信号ＭＣＰＵ－ｒｏｍａを出力してＭＣＰＵ外部メモリアクセス用アドレスラッチ３０Ｍにアドレス情報をラッチさせ、同様にＳＣＰＵ２０はアドレスバスＳＡにアドレス情報を送出し、信号ＳＣＰＵ－ｒｏｍａを出力してＳＣＰＵ外部メモリアクセス用アドレスラッチ３０Ｓにアドレス情報をラッチさせる。同時に発生するＭＣＰＵ－ｒｏｍａ信号とＳＣＰＵ－ｒｏｍａ信号により、メモリ装置競合回避回路５０のＭＣＰＵアクセス要求セットリセット回路５０２とＳＣＰＵアクセス要求セットリセット回路５０４は同時にセットされる。これに対し、上述したＭＣＰＵアクセス優先論理に従い、ＭＣＰＵアクセス実行セットリセット回路５０６がただちにセット状態に変化し、それにより外部メモリデータアクセス制御信号発生回路５１０が外部メモリ９０に対するＭＣＰＵ１０のアクセスオペレーションを実行する。この時点でアドレス切り換え回路４０はＭＣＰＵ１０からのアドレス情報を選択している。ＭＣＰＵ１０のアクセスオペレーションの期間を第２７図の左方の期間ℓで示す（なお、回路

５１０は２相のマスタークロックＣＫ１、ＣＫ２で動作するが、第２６図では図示を省略してある）。外部メモリデータアクセス制御信号発生回路５１０は期間ｎでチップイネーブル信号ＣＥをローアクティブにし、期間ｎの後半の期間ｍで出力イネーブル信号ＯＥをローアクティブする。したがって、この期間ｍにおいて外部メモリ９０からＭＣＰＵ１０が要求したデータが出力され、この期間ｍ内に外部メモリデータアクセス要求信号発生回路５１０から発生する信号ＭＤＬにより、この出力データがＭＣＰＵ外部メモリデータラッチ８０Ｍにラッチされる。これにより、外部メモリデータアクセス要求信号発生回路５１０のＭＣＰＵ１０のためのアクセスオペレーションは完了するので、回路５１０はエンド信号ＥＮＤを出力する。これにより、ＭＣＰＵアクセス実行セットリセット回路５０６はリセットされ、代りにＳＣＰＵアクセス実行セットリセット回路５０８がセットされる。これにより信号ＭＳＥＬはＳＣＰＵアドレス選択を示す"Ｌ"レベルに変化し、アドレス切り換え回路４０はＳＣＰＵ２０からのアドレスを選択して外部メモリ９０をアドレッシングする。更に、ＳＣＰＵアクセス実行セットリセット回路５０８からのセット信号に応答して外部メモリデータアクセス制御信号発生回路５１０がＳＣＰＵ２０のためのアクセスオペレーションを実行する。この期間を第２７図の右側の期間Ｌで示す。このオペレーションにおいて外部メモリデータアクセス制御信号発生回路５１０は信号ＣＥをローアクティブにし、その後半の期間ｐで信号ＯＥをローアクティブにしてＳＣＰＵ２０の要求したデータを外部メモリ９０から出力させ、その出力中に信号ＳＤＬを発生してＳＣＰＵ外部メモリデータラッチ８０ＳにＳＣＰＵ２０の要求したデータをラッチさせる。これにより、外部メモリデータアクセス制御信号発生回路５１０のＳＣＰＵ２０のためのアクセスオペレーションは完了するので同回路５１０はエンド信号ＥＮＤを出力してＳＣＰＵアクセス実行セットリセット回路５０８をリセット状態に戻す。

これ以降、ＭＣＰＵ１０とＳＣＰＵ２０はそれぞれデータバスＭＤ、ＳＤに乗っている外部メモリデータラッチ８０Ｍ、８０Ｓの出力データを読むことにより、要求したデータを得ることができる。

このようにして各ＣＰＵ１０、２０はｒｏｍａ命令（外部メモリアクセス要求命令）を実行後、メモリ装置競合回避回路５０が同ＣＰＵのためのアクセスオペレーションを実行する所定の期間２Ｌだけ待てば要求したデータを得ることができ、アクセス競合の問題が解消される。更に、待機時間が一定（２Ｌ）なので、各ＣＰＵ１０、２０はこの期間を他の命令の実行に使用することができ、プログラム命令の実行効率が最適化される。

なお、ＭＣＰＵ－ｒｏｍａ信号とＳＣＰＵ－ｒｏｍａ信号のタイミング関係がその他のタイミング関係となる場合については図示を省略しているが、いかなる場合でも、各ＣＰＵ１０、２０はｒｏｍａ命令を実行後、所定の期間２Ｌ待てばその時点で既に各ＣＰＵの外部データラッチには要求したデータがラッチされているので、そのデータの入手が可能である。

＜アドレス・データ変換ハードウェア（第２８～第３２図、第１図）＞

一般に、ＣＰＵを含むマイクロコンピュータシステムにおいて、データメモリにある原データから演算用メモリ上に原データを変換したデータ（原データから抽出される所望の情報）を作成することがしばしば望まれる。特にこの種のデータは変換はデータメモリの記憶容量を効率的に使用したような場合にその補償として必要になる。この目的のため、従来では、データメモリから演算用メモリへの転送命令を実行して、データメモリの原データを演算用メモリに移し、次に１以上の変換命令を実行して、演算用メモリにあるデータをＡＬＵを介して変換する。したがって、従来の場合、演算用メモリ上に所望のデータを得るためのデータ変換手続に時間がかかり、高速処理が要

求されるアプリケーションにおける障害の1つとなっている。

本実施例ではCPU10、20がデータメモリである外部メモリ90から演算用メモリである内部RAM106または206にデータを転送する命令（roma命令）を実行するだけで、所望の変換が施されたデータが内部RAM106、206に読み込まれるようにして、データ変換処理の高速化を図っている。この目的を実現するため、CPU10、20と外部メモリ90との間のアドレス経路上にアドレス変換回路60が設けられ、また外部メモリ90とCPU10、20との間のデータ経路上にデータ変換回路70が設けられ、各変換回路60、70はroma命令の実行時にCPU10、20から与えられる制御信号に応答して所望の変換を実行する。

第28図に外部メモリアクセス命令romaのリストを示す。第1の命令roma0は変換なしの転送命令であり、これに対し、アドレス変換回路60はCPU10、20から与えられる入力アドレスをそのまま出力アドレスとして外部データメモリ90に通し、データ変換回路70も外部データメモリ90からのデータ（16ビットデータ）を無変換で通してCPU10、20に渡す。この無変換転送命令roma0ではCPU10、20から変換回路60、70に与えられる変換制御用の信号R1、R2、R3はいずれも"L"レベルとなる。

第2の命令roma1は特殊波形の読み出しに適した命令である。この命令に対し、アドレス変換回路60はCPU10、20から送られてきた入力アドレスの第13ビットA12が"0"のときは下位12ビットを無変換で通すが第13ビットA12が"1"のときは下位12ビットを反転させる。なお、アドレス変換回路60の出力アドレスの第13ビットは入力アドレスの第13ビットA12の値にかかわらず"0"に固定される。また、この命令に対し、データ変換回路70はCPU10、20から送られてきた入力アドレスの第13ビットA12をCPU10、20に送るデータの第13ビットD12とするとともにA12が"1"のとき下位の12ビットデータを反転する形式で外部メモリ90からのデータを変換する。したがって、外部メモリ90のアドレス領域0000～0FFFに第28図に示すような有効データビット数12の特殊波形データ（0000～0FFF）があるとすると、CPU10、20がこの命令を指定アドレス0000～1FFFの範囲について繰り返し実行した場合に、アドレス変換回路60から出力される外部メモリアドレスはいったん0000から0FFFに進み、この間、データ変換回路70は外部メモリ90からのデータをそのまま通し、その後、アドレス変換回路60の反転動作により、外部メモリ90へのアドレスは0FFFから0000に後退し、この間、データ変換回路70は外部メモリ90から出力されるデータの下位12ビットを反転し、第13データビットD12を"1"にして変換されたデータを出力する。結局、CPU10、20がアドレスを0000～1FFFに動かして命令roma1を繰り返し実行した場合に、CPU10、20が実際に受け取る波形は第28図のroma1の欄の右方に示すような波形となる。この変換波形は左方に示す外部メモリ90内の原波形を所定の態様で延長した繰り返し波形（アドレス0FFF、データ0FFFの点について対称な波形）である。この結果、記憶容量の点についていうと、変換波形のデータ自体を予め外部データメモリ90に記憶させる方式に比べ、波形データ記憶容量が半分になる利点がある。この命令roma1の場合、制御信号R1、R2、R3のうちR1のみが"H"レベルになる。

第3の命令ROMA2は外部メモリデータの一部（半語）の読み出しを指示する命令である。この命令の場合、R2のみが"H"レベルになる。外部データメモリ90の1アドレス（1語）当りの記憶容量は16ビットである。この命令roma2に対し、データ変換回路70は、CPU10、20からのアドレスの第16ビットA15が"0"のときは、外部データメモリ90からの1

5ビットデータのうち、下位の8ビットを残し、上位の8ビットを"0"にマスクする変換を実行し、A15が"1"のときは外部データメモリ90から16ビットデータのうち、上位の8ビットを下位8ビットにシフトする（残った上位8ビットはマスク）変換を実行する。また、データ変換回路70において入力アドレスの第16ビットA15を制御信号として使用しているので、アドレス変換回路60ではA15の値にかかわらず出力アドレスの第16ビットを所定値"0"にマスクする。なお、この場合において外部データメモリ90からの16ビット情報の上位8ビットと下位ビットとの関係は、1つのデータ（例えば位相データ）における上位データ部分（例えば整数部）と下位データ部分（例えば小数部）のような関係であってもよいし、異なる2種類の8ビットデータ（例えばレートデータとレベルデータ）の各々であるような独立な関係であってもよい。

第4の命令ROMA3は外部メモリデータをシフトして一部を読み出す命令である。この命令の場合、R3のみが"H"レベルになる。この命令に対し、データ変換回路70は外部メモリ90からの16ビットデータのうち、bit15はそのままにして上位12ビットのbit15～bit4をbit14～bit3にシフトし、下位の3ビットbit2～bit0を0にマスクする変換を行う。ここに、外部メモリ90の16ビットデータのうち上位12ビットは例えばbit15を符号ビットとする波形データであり、下位4ビットは別のデータを表わす。この場合、上記の変換により、CPU10、20は内部RAM106、206上で使用するのに適したフォーマットの波形データを高速に読み取ることができる。

第29図にアドレス変換回路60のブロック図を示す。このアドレス変換回路60にはMCPU10またはSCPU20からアドレスラッチ30M、30S、アドレス切り換え回路40を介して入力される16ビットのアドレスのうち、下位12ビット（bit0～bit11）が詳細を第30図に示す反転回路610に入力される。この反転回路610は信号R1が命令roma1を表わす"1"でアドレスのA12が"1"のときANDゲート612からの信号により動作して入力されるアドレスの下位12ビットを反転させる。また、命令roma1の実行時に"1"となる信号R1はインバータ602を介して、ANDゲート604を禁止し、入力アドレスのA12の値にかかわらず出力アドレスの対応ビット（bit12）を"0"にする。入力アドレスのA13とA14はそのまま出力アドレスの対応ビット（bit13、bit14）として出力される。入力アドレスのA15（MSB）はANDゲート608を介して出力アドレスの対応ビット（bit15）となる。命令roma2の実行中を表わす"1"の信号R2が発生しているとき、この信号R2がインバータ606を介してANDゲート608を禁止して出力アドレスのbit15（MSB）を"0"にマスクする。

したがってアドレス変換回路60は、無変換命令roma0とシフト読み出し命令roma3に対してはR1="0"、R2="0"なので入力アドレスを出力アドレスとしてそのまま通し、特殊波形読出し命令roma1に対してはR1="1"なので出力アドレスのbit12を"0"にマスクし、A12="1"の間反転回路610により入力アドレスの下位12ビット（bit0～bit11）を反転して出力アドレスとする。更に、一部読み出し命令roma2に対してはR2="1"なので出力アドレスのbit15を"0"にマスクする。このようにして、第28図に関して述べたアドレス変換回路の機能が実現される。

第31図にデータ変換回路70のブロック図を示し、第32図にその詳細を示す。これらの図においてデータ入力は第1図の外部メモリ90から供給されるデータである。第32図において、入力データの上位8ビットに結合する3状態ゲート回路702と入力データの下位8ビットに結合する3状態ゲート回路704は出力するデータの下位8ビットとして入力データの上位8ビットを選

択するが、入力データの下位８ビットを選択するかを決めるためのものである。Ｒ２＝“１”（ｒｏｍａ２命令）でＡ１５＝１のとき、ＡＮＤゲート７０５の“１”出力信号とその反転信号であるインバータ７０８の出力信号“０”により、ゲート回路７０２が導通し、ゲート回路７０４がオフして入力データの上位８ビットが出力データの下位８ビットとして選択される。その他の場合は、ゲート回路７０２がオフし、ゲート回路７０４が導通するので入力データの下位８ビットがそのまま出力データの下位８ビットとして出力される。更にＲ２＝“１”（ｒｏｍａ２命令）のときは、入力データの上位８ビットに結合するＡＮＤゲート回路７１０が禁止されて出力データの上位８ビットを“０”にマスクする。即ち、Ｒ２＝“１”のときはインバータ７１２とＮＯＲゲート７１４を介して禁止信号がＡＮＤゲート回路７１０に加わってＡＮＤゲート回路７１０における入力データ上位８ビットの通過が阻止される。また、ＡＮＤゲート回路７１０における入力データの上位３ビットと結合するＡＮＤゲート素子はＲ１＝“１”（ｒｏｍａ１命令）のときにＮＯＲゲート７１４を介して禁止され、出力データの上位３ビットを“０”にマスクする。

ＥＸ－ＯＲゲート回路７１６は入力データの下位１２ビットを選択的に反転するための回路である。ＥＸ－ＯＲゲート回路７１６はＲ１＝“１”（ｒｏｍａ１命令）でＡ１２＝１のとき、ＡＮＤゲート７１８からの反転信号“１”により、下位１２ビットデータを反転し、その他の場合は下位１２ビットデータをそのまま通す。回路７１０内のＡＮＤゲート素子を介して入力データのｂｉｔ１２に結合する状態ゲート７２２はＲ１＝“１”（ｒｏｍａ１命令）のときに、信号Ｒ１に結合するインバータ７２０を介して与えられる信号“０”によりオフし、代りに、Ａ１２に結合する３状態ゲート７２４が信号Ｒ１によって導通して出力データのｂｉｔ１２を発生する。シフトマスク回路７２６は選択的に入力されたデータのｂｉｔ１５～ｂｉｔ４を出力データのｂｉｔ１４～ｂｉｔ３にシフトし、出力データのｂｉｔ２～ｂｉｔ０を“０”にマスクするための回路であり、Ｒ３＝“１”（ｒｏｍａ３命令）のとき信号Ｒ３に結合するインバータ７２８からの信号“１”によってこの変換を実行する。

したがって、データ変換回路７０は、無変換命令ｒｏｍａ０（Ｒ１＝Ｒ２＝Ｒ３＝“０”）のときは、入力される１６ビットデータをそのまま通し、特殊波形読み出し命令ｒｏｍａ１（Ｒ１＝“１”）のときは入力アドレスの上位４ビット（ｂｉｔ１５～ｂｉｔ１２）が“００００”（Ａ１２＝０のとき）か“０００１”（Ａ１２＝１のとき）かによって、出力データの下位１２ビットをそのまま入力データの下位１２ビットとする（Ａ１２＝０のとき）か、或は、出力データの下位１２ビットを入力データの下位１２ビットが反転されたデータとなる（Ａ１２＝１）ようにデータ変換を行い、一部読み出し命令ｒｏｍａ２（Ｒ２＝“１”）のときは出力データの上位８ビットがオールゼロで、出力データの下位８ビットが入力データの下位８ビットとなるように（Ａ１５＝０のとき）、或は、出力データの上位８ビットがオールゼロで、出力データの下位８ビットが入力データの上位８ビットとなる（Ａ１５＝１のとき）ようにデータ変換を行い、シフト読み出し命令ｒｏｍａ３（Ｒ３＝１）のときは出力データの下位３ビット（ｂｉｔ０～ｂｉｔ２）がオールゼロで、出力データのｂｉｔ３～ｂｉｔ１４が入力データのｂｉｔ４～ｂｉｔ１５で、出力データのｂｉｔ１５（ＭＳＢ）が入力データのｂｉｔ１５（ＭＳＢ）となるようにデータ変換を行う。このようにして第２８図で述べたデータ変換機能が達成されている。

以上により、アドレス変換回路６０とデータ変換回路７０とを設けたことによる利点は明らかである。即ち、ＣＰＵ１０、２０にとつて、データメモリである外部メモリ９０に対するアクセス命令ｒｏｍａを実行するだけで、回路６０と７０の変換機能により、所望の変換が施されたデータをただちに得ることができ、従来のように、外部メ

モリ９０のデータを演算用メモリである内部ＲＡＭ１０６、２０６にいったん取り込んだ後に、ＡＬＵ部１０８、２０８のようなＡＬＵを介して変換を実行する必要がなく、処理が高速化される利点がある。

なお、第２８図に示したアクセス命令ｒｏｍａのリストは例示にすぎず、拡張、変更は容易である。

＜ＤＡＣサンプリング（第３３、第３４図）＞

本実施例においてＤＡＣ１００はＭＣＰＵ１０とＳＣＰＵ２０が生成したデジタル楽音信号をアナログ楽音信号に変換するものである。第５図の５－５に示すように、ＭＣＰＵ１０はタイマインタラプト処理ルーチンのなかで、ＭＣＰＵ１０とＳＣＰＵ２０が生成したデジタル楽音信号のサンプルをＤＡＣ１００にセットする。この処理５－５の実行間隔は平均としてはタイマインタラプト発生部１１６の発生するインタラプト信号ＩＮＴの発生間隔に等しいが、実際の実行間隔はプログラム動作のために変動する。したがって、処理５－５の実行間隔をＤ／Ａ変換の変換周期としてＤ／Ａ変換を行ったとするとアナログ楽音信号に大きな歪みが生じてしまう。

第３３図に右ＤＡＣ１００Ｒまたは左ＤＡＣ１００Ｌの構成例を示す。第３３図の（Ａ）に示す構成では、処理５－５の実行時に、ＭＣＰＵ１０のオペレーション制御回路１１２の制御の下に、内部ＲＡＭ１０６内の波形加算用レジスタが指定され、そこに記憶されている最新のデジタル楽音データが取り出され、データバスに乗せられる。そして、データバスにデジタル楽音データが乗っているタイミングでラッチ１００４のクロック入力にストローブ用のプログラム制御信号がオペレーション制御回路１１２から与えられデータバス上のデータがセットされ、ラッチ１００４から新しいデジタル楽音データがＤ／Ａ変換器１００２に入力される。したがって、第３４図（Ａ）に示すように、Ｄ／Ａ変換器１００２に入力されるデジタル楽音データはプログラム制御のために不安定な周期で切り換わることになる。Ｄ／Ａ変換器１００２の変換周期（サンプリング周期）は非常に安定していなければ、その変換において大きな歪みが発生する。

この問題は第３３図（Ｂ）に示すような構成をとることにより解決される。すなわち、オペレーション制御回路１１２からのプログラム制御信号によって制御されるソフト制御ラッチ１００４と、デジタル楽音信号をアナログ楽音信号に変換するＤ／Ａ変換器１００２との間に、インタラプト発生部１１６からの正確なタイミング信号であるインタラプト信号ＩＮＴで制御されるインタラプト制御ラッチ１００６を設ける。インタラプト信号の発生周期はクロック発振器の安定度に従うので極めて安定である。ラッチ１００６の出力はインタラプト信号のタイミングに同期して切り換わる。すなわち、インタラプト信号の発生周期がＤ／Ａ変換器１００２の変換（サンプリング）周期となる。第３３図（Ｂ）の構成に対するタイムチャートを第３４図（Ｂ）に示す。図示のように、ラッチ１００４の出力が切り換わるタイミングはインタラプト処理に移行するタイミングのずれや、該インタラプト処理に要する時間（斜線部の長さ）によって変動するがインタラプト信号で動作するラッチ１００６があるのでＤ／Ａ変換器１００２の入力データが切り換るタイミングはインタラプト信号と同期する。これにより、第３３図（Ａ）の構成における歪み問題が解決される。

［変形例］

以上で実施例の説明を終えるがこの発明の範囲内で種々の変形、変更が可能である。

この発明は複数のＣＰＵを有するデジタルマイクロコンピュータにおけるＣＰＵ間のアクセス技術に関する。したがって、この発明はＣＰＵ間でアクセスが行われる任意のデジタルマイクロコンピュータに適用し得、特定のアプリケーション（実施例では電子楽器）には制限されない。

実施例ではＣＰＵ間のアクセスが１つのＣＰＵ

（ＭＣＰＵ）により一方向でのみ行われる環境（ＭＣＰＵ１０はＳＣＰＵ２０をアクセスするがＳＣＰＵ２０はＭＣＰＵ１０をアクセスしない）を想定しているが、ＣＰＵ間のアクセスが両方向で行われる環境や複数のＣＰＵが同じＣＰＵをアクセスする環境にもこの発明を適用し得る。例えば、デュアルＣＰＵマイクロコンピュータにおいてこれを実現するためには、第１と第２のＣＰＵの各々を独自のクロック発生回路で動作するようにし、各クロック発生回路に実施例で述べたような停止／再開機能を設ける。また各ＣＰＵの内部ＲＡＭに対するアクセス径路として自身のＣＰＵからのアクセス径路と外部のＣＰＵからのアクセス径路とを設け、外部のＣＰＵからの制御信号（下記のアクセス調停回路を通した信号）によってアクセス径路の選択を行うようにする。第１と第２のＣＰＵが同時に相手のＣＰＵに対してアクセスを試みる場合を想定して、各ＣＰＵ（第１のＣＰＵ、第２のＣＰＵ）からのアクセス要求信号に応答するアクセス調停回路を設ける。アクセス調停回路は、第１のＣＰＵが第２のＣＰＵをアクセスしている間は、第２のＣＰＵが第１のＣＰＵをアクセスしないようにし、逆に第２のＣＰＵが第１のＣＰＵをアクセスしている間は第１のＣＰＵが第２のＣＰＵをアクセスしないように両ＣＰＵのアクセスを制御する。このために例えば、アクセス調停回路は第１と第２のＣＰＵから同時にアクセス要求信号を受けた場合に、優先ロジックに従い例えば第１のＣＰＵを優先させ、出力信号として第１のＣＰＵからのアクセス要求信号（"Ｈ"）は第２のＣＰＵに送る（とともに、第１のＣＰＵからのアクセスが可能なことを第１のＣＰＵのオペレーション制御回路に知らせる）が、第２のＣＰＵからのアクセス要求信号は禁止レベル（"Ｌ"）にして第１のＣＰＵに送る（とともに第２のＣＰＵからのアクセスができないことを第２のＣＰＵのオペレーション制御回路に知らせる）。更に、アクセス調停回路は片方のＣＰＵからのアクセス要求信号を受信し、その信号を他方のＣＰＵのクロック発生回路と内部メモリのアクセス径路の選択回路、及び片方のＣＰＵのオペレーション回路に送っている間に、他方のＣＰＵからアクセス要求信号を受信したような場合（アクセスオペレーションがオーバーラップするような場合）他方のＣＰＵからのアクセス要求信号は禁止レベルにして各ＣＰＵに出力する。

各ＣＰＵにおいてアクセスが実行できるかどうかを検出するために、アクセス調停回路からの出力信号を各ＣＰＵのオペレーション制御回路で受ける。各ＣＰＵのオペレーション制御回路は相手のＣＰＵをアクセスする命令を実行するオペレーションの最初のステップで、アクセス調停回路からの出力信号を読み、出力信号がアクセス許可レベル（"Ｈ"）の場合にはそのままアクセスオペレーションを実行するが、出力信号がアクセス禁止レベル（"Ｌ"）の場合には、今回の相手のＣＰＵへのアクセスオペレーションを取り止め、次のマシンサイクルで再び相手のＣＰＵへのアクセスを試みるために、ＲＯＭアドレス制御部を制御して、次のマシンサイクルで再び制御用ＲＯＭから相手のＣＰＵをアクセスするための命令語が取り出されるようにする。

結果として両ＣＰＵが同時にアクセスを試みる場合を除いて、各ＣＰＵはアクセス命令を１回実行するだけで、相手のＣＰＵをアクセスすることができる。また、同時アクセスのために、アクセスの要求が受け付けられなかったＣＰＵも、再度、アクセス命令を実行することで相手のＣＰＵをアクセスできる。したがって、ＣＰＵ間で両方向のアクセスを必要とする環境や複数のＣＰＵが同じＣＰＵをアクセスするような環境でも、ＣＰＵ間のアクセスを最適化することができる。

［発明の効果］

以上、詳細に説明したように、この発明によれば、複数のＣＰＵを有するデジタルマイクロコンピュータにおけるＣＰＵ間のアクセスを、アクセスが要求されるＣＰＵが動作中のときでもアクセスを要求するＣＰＵがアクセス命令を実行するだけで高速に達成することができ、従来のように、

時間のかかるアクセス手続を踏む必要がない。したがって、システムとしてのデジタルマイクロコンピュータの動作効率が上り、その性能の向上が図られる。

4、図面の簡単な説明

第1図はこの発明を適用した電子楽器用処理装置の全体構成図、

第2図は第1図のMCPUのブロック図、

第3図は第1図のSCPUのブロック図、

第4図はMCPUの実行するメインプログラムのフローチャート、

第5図はMCPUの実行するインタラプト処理ルーチンのフローチャート、

第6図はSCPUの実行するプログラムのフローチャート、

第7図は音源処理のフローチャート、

第8図は時間の経過に沿う実施例の動作のフローチャート、

第9図はチャンネル音源処理のフローチャート、

第10図は波形データを示す図、

第11図は音源処理用RAMテーブルを示す図、

第12図はSCPU動作開始終了機能に関係する回路のブロック図、

第13図、第14図、第15図は第12図の回路の動作のタイムチャート、

第16図はインタラプトマスク機能を有する回路のブロック図、

第17図はインタラプトマスク方式によるエンベロープ設定処理のフローチャート、

第18図は単一命令で複数のデータを転送する間インタラプト信号によるメインプログラムの中断を禁止する機能を有する回路のブロック図、

第19図は複数のデータを単一命令で転送するのに適したRAMのメモリマップ例を示す図、

第20図は複数の転送命令による動作と単一の転送命令による動作とを比較して示す図、

第21図は単一転送命令方式によるエンベロープ設定処理のフローチャート、

第22図はSCPUの停止モード利用によるMCPUからのSCPUアクセス機能を説明するのに用いたフローチャート、

第23図はSCPUに対する瞬時強制アクセス機能を有するMCPUのブロック図、

第24図はSCPUに対する瞬時強制アクセス機能に適合するSCPUのブロック図、

第25図はMCPUからSCPUの内部RAMにデータを書き込む場合の動作のタイムチャート、

第26図は第1図のメモリ装置競合回避回路のブロック図、

第27図は第26図の回路の動作のタイムチャート、

第28図は外部メモリからのデータを変換して取り込む命令を含む外部メモリアクセス命令のリストを示す図、

第29図は第1図のアドレス変換回路のブロック図、

第30図は第29図の反転回路の回路図、

第31図は第1図のデータ変換回路のブロック図、

第32図はデータ変換回路の回路図、

第33図は第1図のDACのサンプリング周期が不安定になる構成とサンプリング周期を安定化した構成とを比較して示す図、

第34図はDACのサンプリング周期が不安定な場合のタイムチャートと安定な場合のタイムチャートとを比較して示す図である。

10……MCPU（第1のCPU）

20……SCPU（第2のCPU）

112M……オペレーション制御回路（アクセス要求信号発生手段、アクセス実行手段、アクセス終了信号発生手段）

D……（ハイレベルの変化がアクセス要求信号を表わし、ローレベルへの変化がアクセス終了信号を表わす）

136……クロック発生回路（第1のクロック

発生手段）

235……クロック発生回路（オペレーション中断手段、オペレーション再開手段、第2のクロック発生手段、クロック停止手段、クロック再起動手段）

128……アドレスパスゲート

204……RAMアドレス制御部

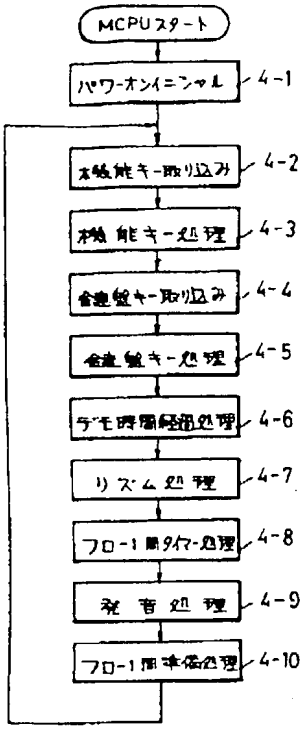
240……RAMデータ切り換え部

242……ライト信号切り換え部

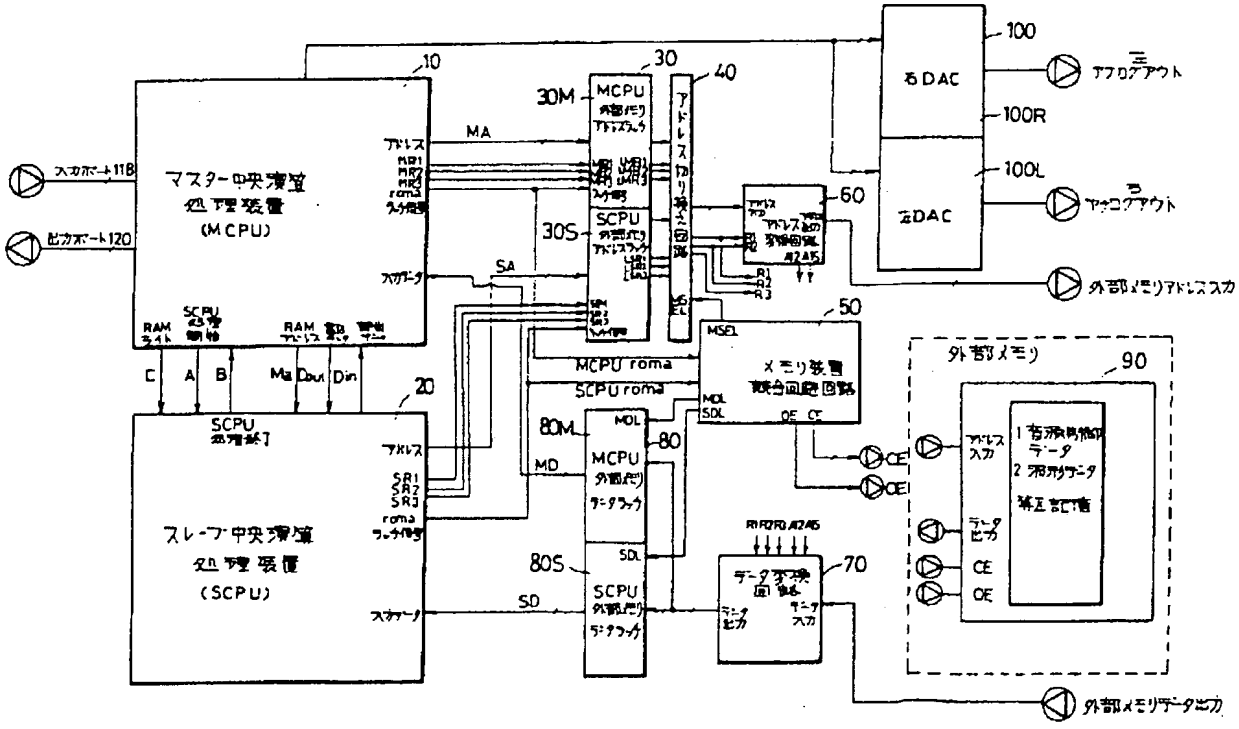
128、204、240、242……（オペレーション中断手段、アクセス径路切換手段、アクセス径路復帰手段）

特許出願人　カシオ計算機株式会社

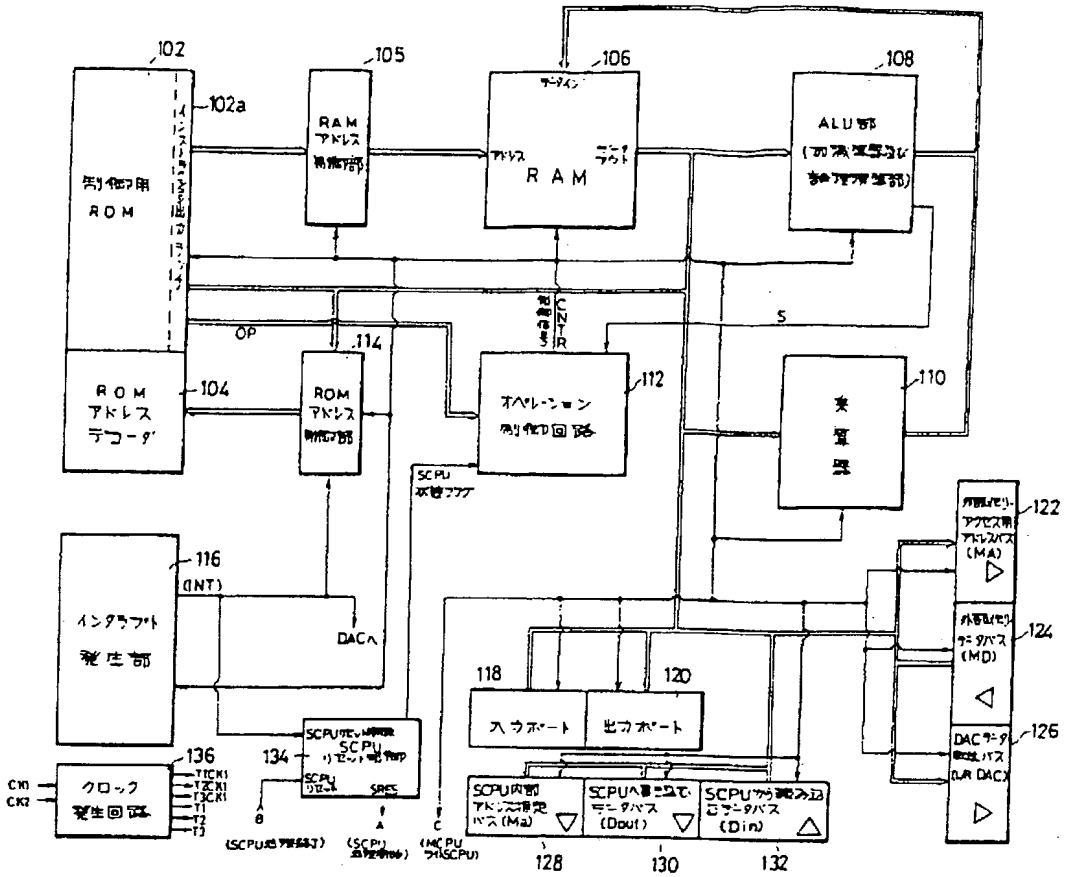
代理人　弁理士　長南満輝男



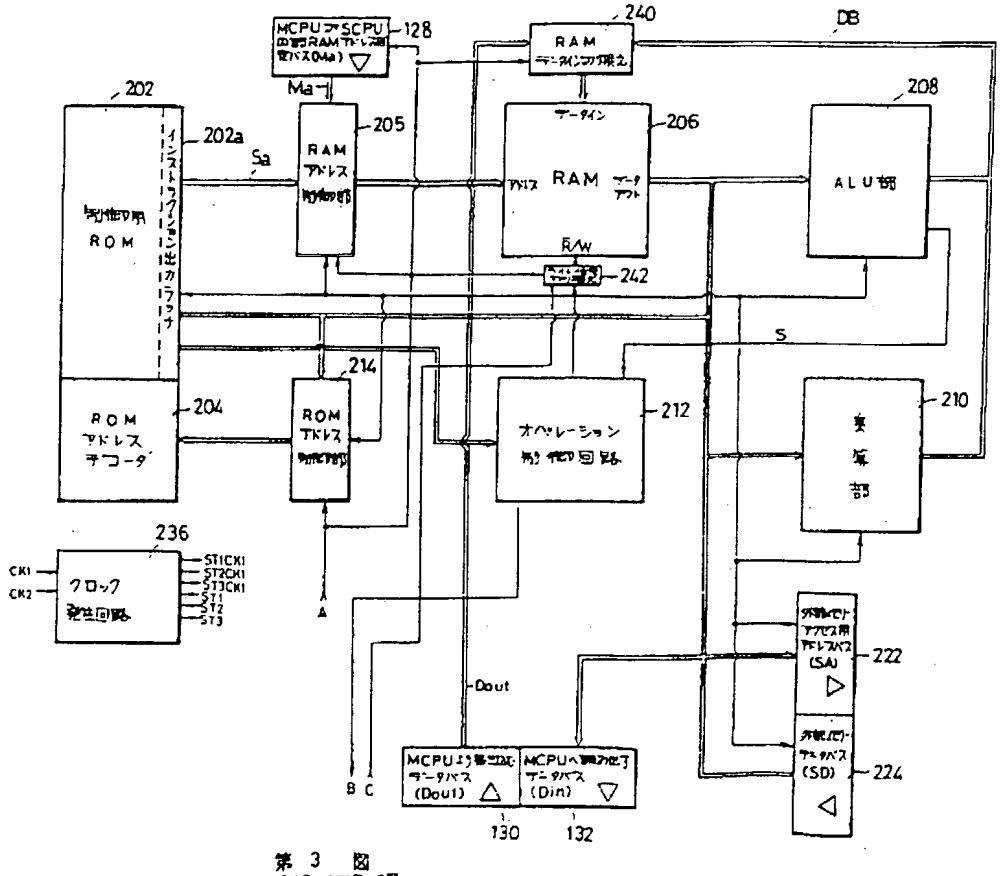
第4図



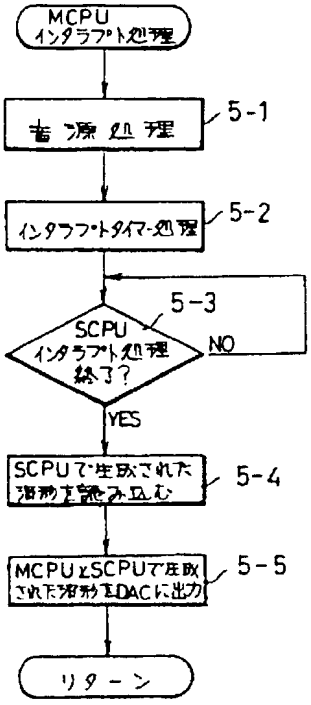
第1図
全体構成



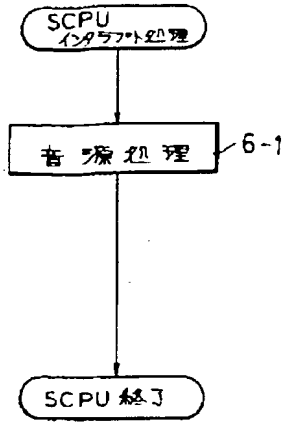
第 2 図
MCPUのブロック図



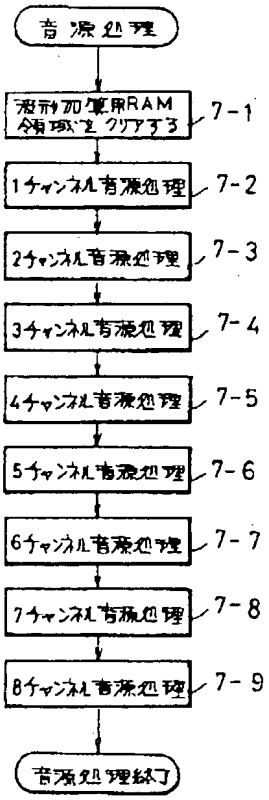
第 3 図
SCPUのブロック図


MCPU
インタラプト処理
音源処理
5-1
インタラプトタイマ-処理
5-2
5-3
SCPU
インタラプト処理
終了?
NO
YES
SCPUで生成された
波形を読み込む
5-4
MCPUとSCPUで生成
された波形をDACに出力
5-5
リターン

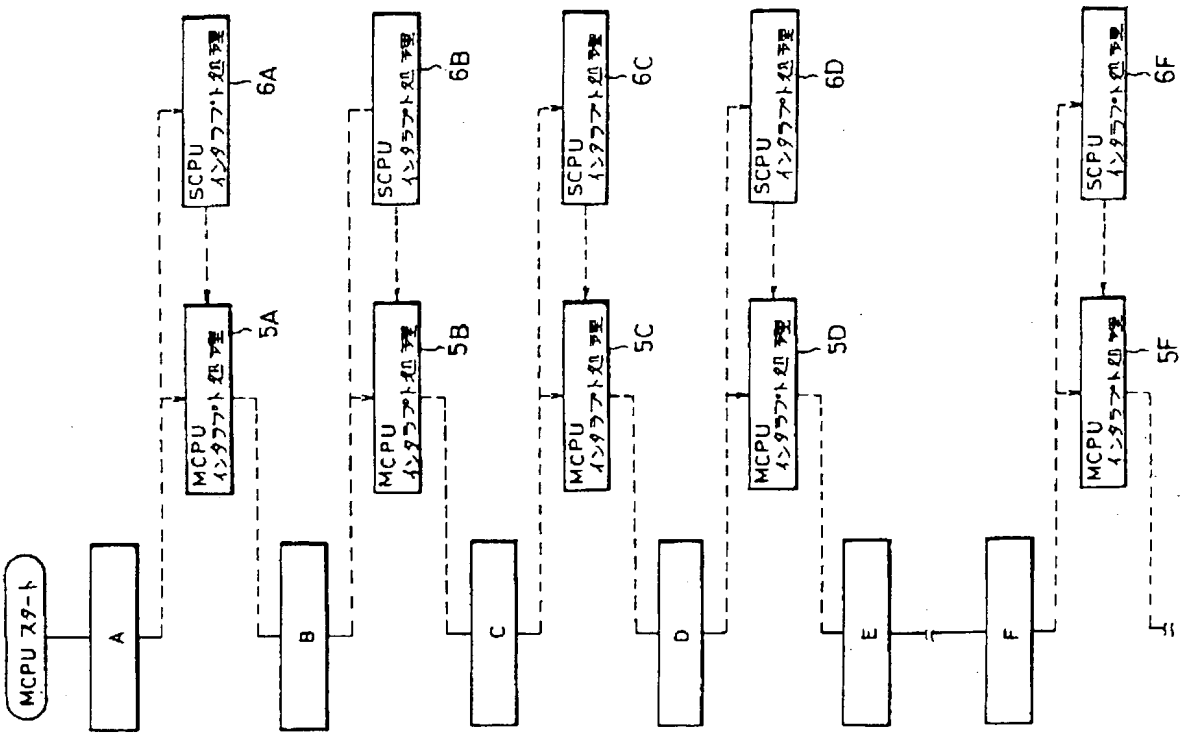
第 5 図


SCPU
インタラプト処理
音源処理
6-1
SCPU終了

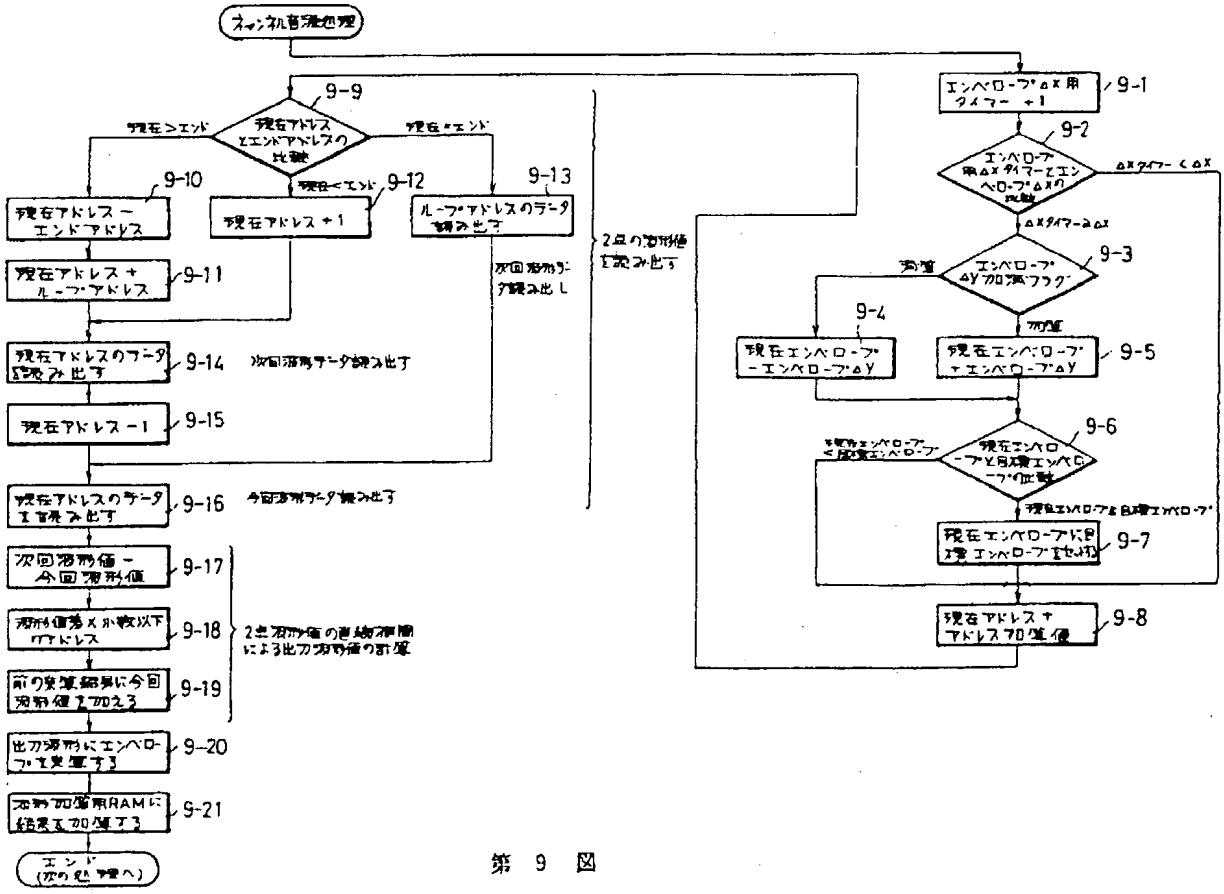
第 6 図


音源処理
波形加算用RAM
領域をクリアする
7-1
1チャンネル音源処理
7-2
2チャンネル音源処理
7-3
3チャンネル音源処理
7-4
4チャンネル音源処理
7-5
5チャンネル音源処理
7-6
6チャンネル音源処理
7-7
7チャンネル音源処理
7-8
8チャンネル音源処理
7-9
音源処理終了

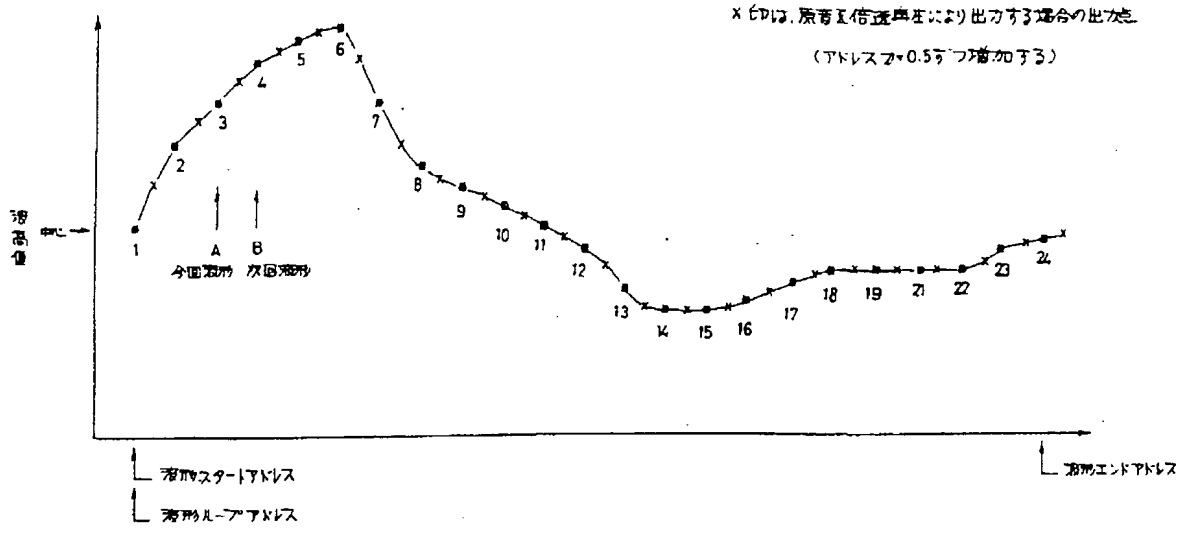
第 7 図


MCPU スタート
A
B
C
D
E
F
MCPU インタラプト処理
SCPU インタラプト処理
5A
6A
5B
6B
5C
6C
5D
6D
5F
6F

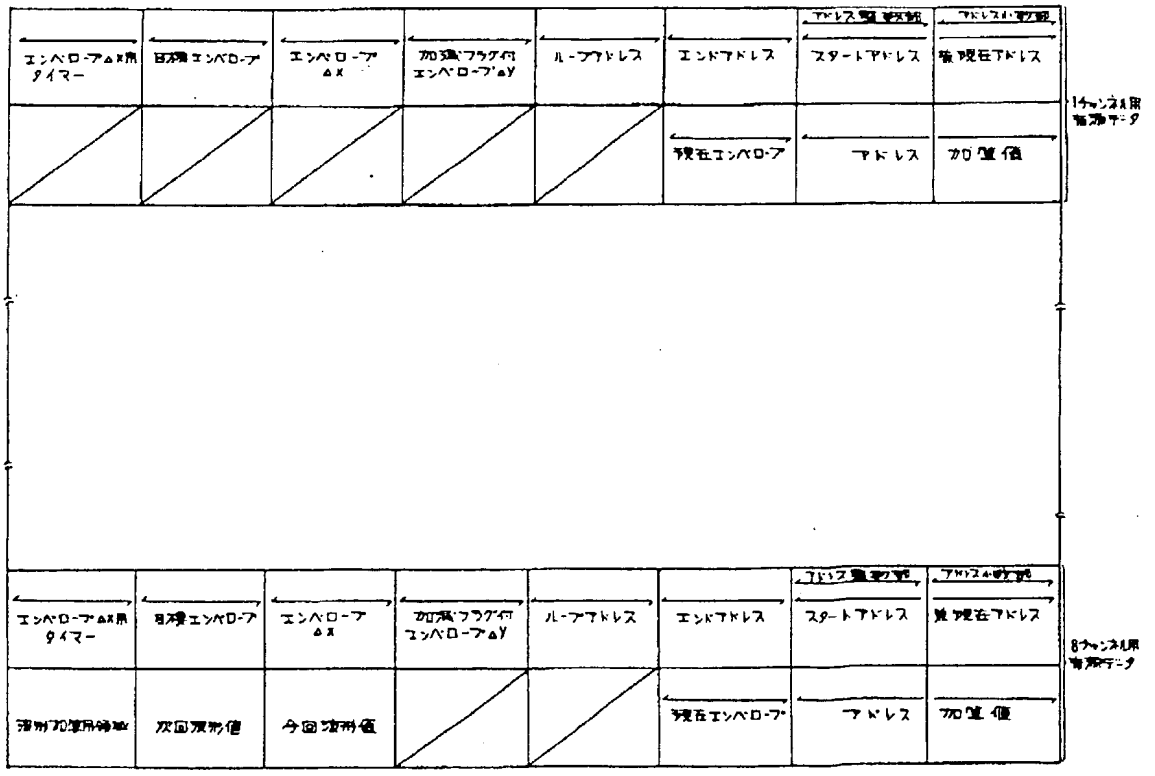
第 8 図


オッシル音源処理
9-9
現在アドレス エンドアドレスの 比較
現在＞エンド
現在＝エンド
現在＜エンド
9-10
現在アドレス − エンドアドレス
9-12
現在アドレス＋1
9-13
ループアドレスのデータ 読み出す
9-11
現在アドレス ＋ ループアドレス
次回波形データ読み出し
9-14
現在アドレスのデータ 読み出す
次回波形データ読み出す
9-15
現在アドレス − 1
9-16
現在アドレスのデータ 読み出す
今回波形データ読み出す
2点の波形値 読み出す
9-17
次回波形値 − 今回波形値
9-18
波形値差×小数以下 アドレス
2点波形値の直線補間 による出力波形値の計算
9-19
前の演算結果に今回 波形値を加える
9-20
出力波形にエンベロープを乗算する
9-21
波形加算用RAMに 結果を加算する
エンド (次の処理へ)
エンベロープΔX用 タイマー＋1
9-1
9-2
エンベロープ用ΔXタイマーとエンベロープΔXの比較
ΔXタイマー＜ΔX
ΔXタイマー＝ΔX
9-3
エンベロープ ΔY加減フラグ
減算
加算
9-4
現在エンベロープ − エンベロープΔy
9-5
現在エンベロープ ＋ エンベロープΔy
9-6
現在エンベロープと目標エンベロープの比較
現在エンベロープ＜目標エンベロープ
現在エンベロープ＝目標エンベロープ
9-7
現在エンベロープに目標エンベロープを代入
9-8
現在アドレス ＋ アドレス加算値

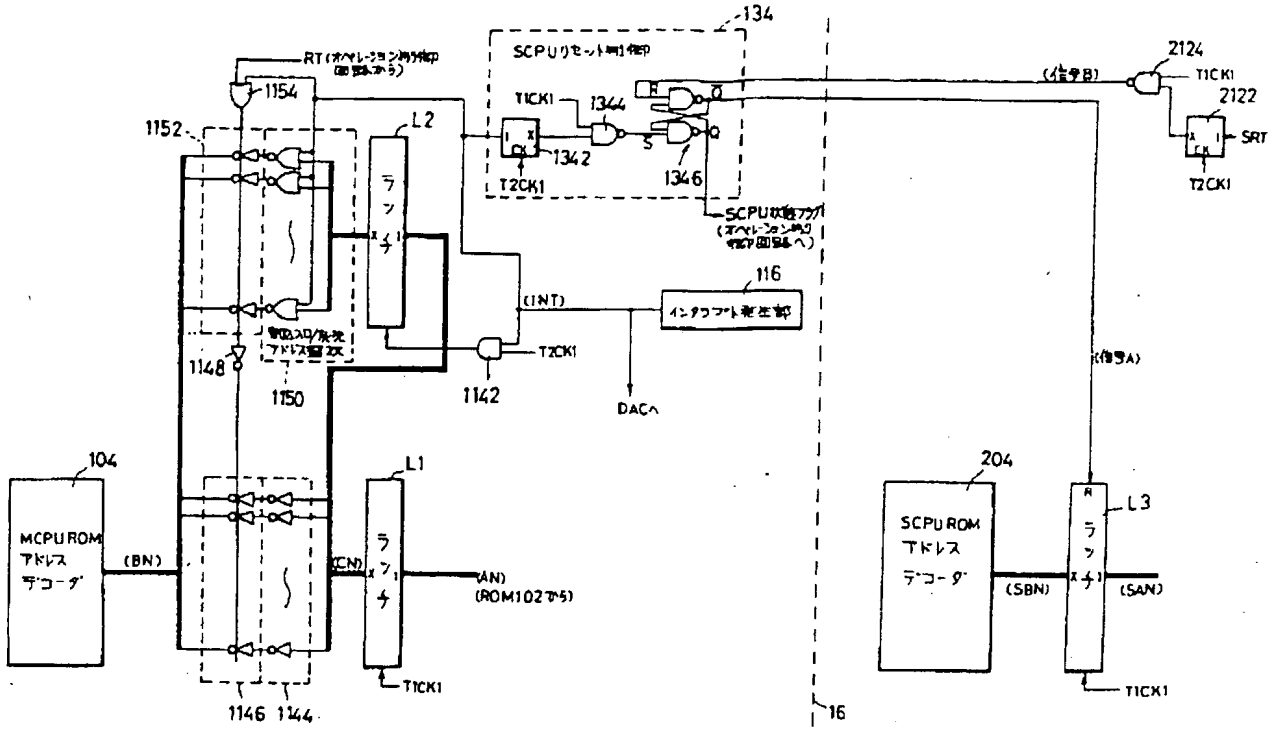
第 9 図



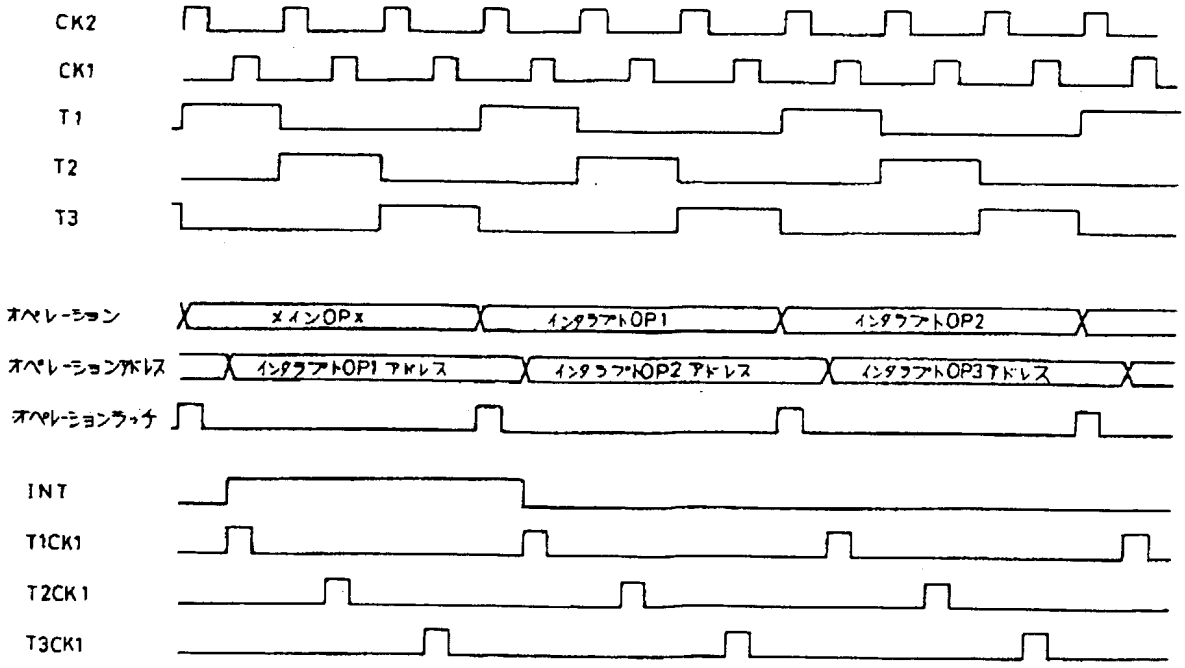
第 10 図
波 形 デ ー タ



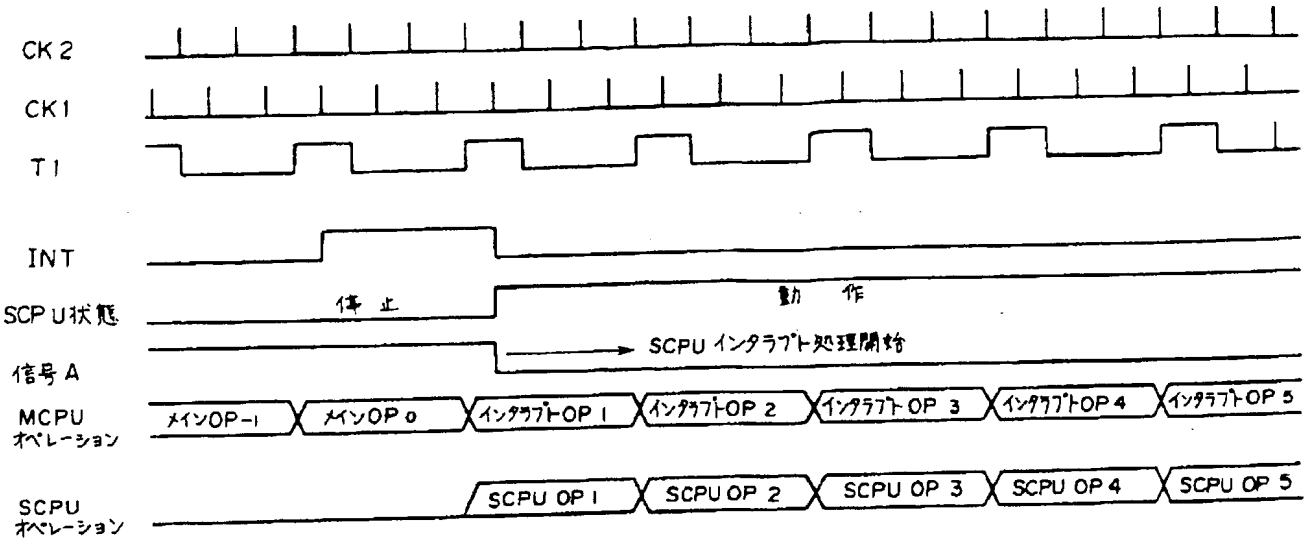
第 11 図
音源処理用RAMテーブル


134
SCPUリセット制御部
RT(オペレーション終了信号)
1154
1152
L2
T1CK1
1344
1342
T2CK1
1346
SCPU起動フラグ
2124
T1CK1
2122
SRT
T2CK1
116
(INT)
インタラプト発生部
T2CK1
1142
DACへ
1148
1150
104
MCPU ROM
アドレス
デコーダ
(BN)
L1
(CN)
(AN)
(ROM102から)
T1CK1
1146
1144
204
SCPU ROM
アドレス
デコーダ
(SBN)
L3
(SAN)
T1CK1
16

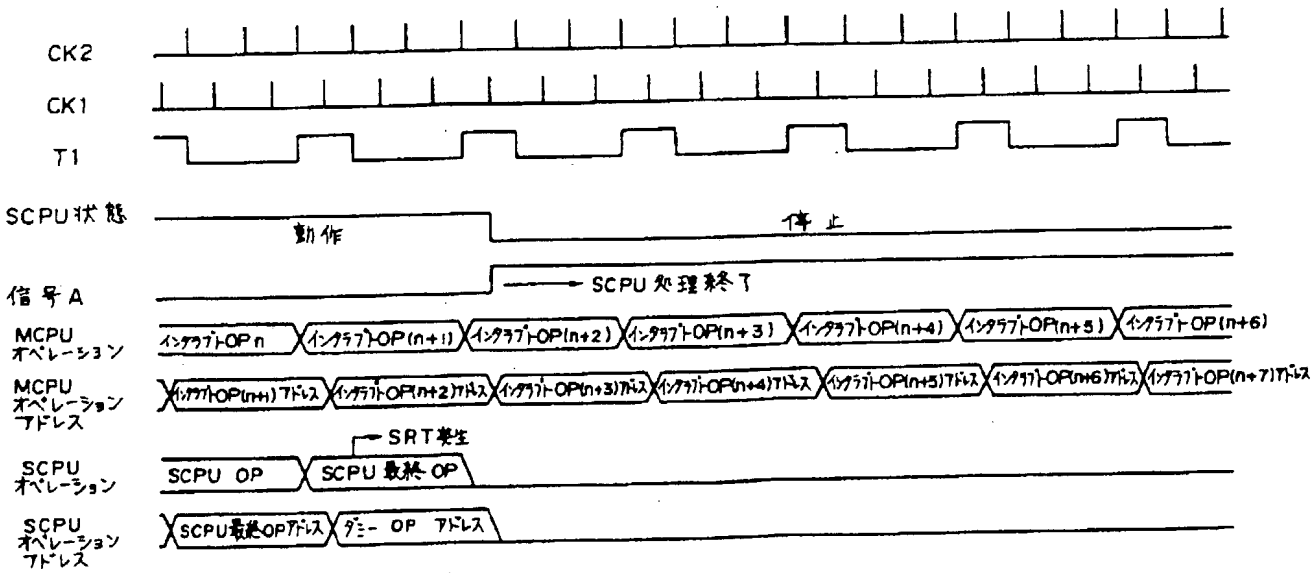
第 12 図


CK2
CK1
T1
T2
T3
オペレーション
メインOPx
インタラプトOP1
インタラプトOP2
オペレーションアドレス
インタラプトOP1 アドレス
インタラプトOP2 アドレス
インタラプトOP3アドレス
オペレーションラッチ
INT
T1CK1
T2CK1
T3CK1

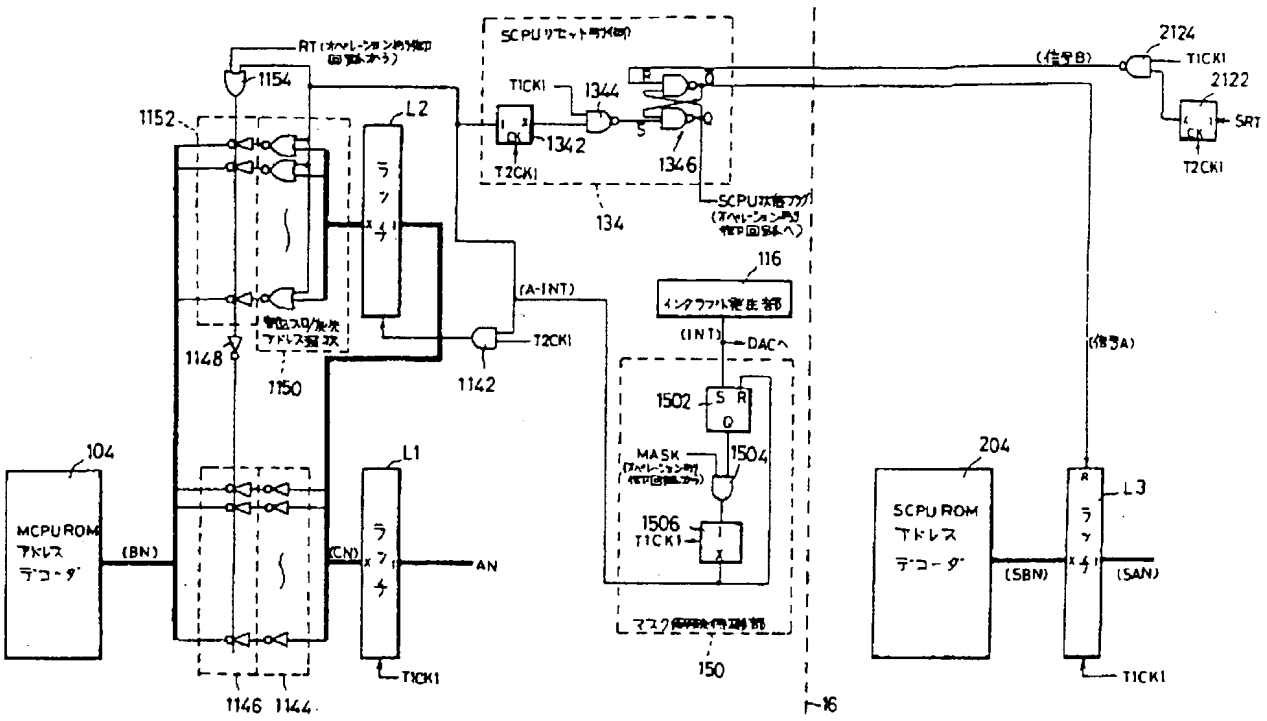
第 13 図


CK2
CK1
T1
INT
SCPU状態
停止
動作
信号A
SCPUインタラプト処理開始
MCPU
オペレーション
メインOP-1
メインOP 0
インタラプトOP 1
インタラプトOP 2
インタラプトOP 3
インタラプトOP 4
インタラプトOP 5
SCPU
オペレーション
SCPU OP 1
SCPU OP 2
SCPU OP 3
SCPU OP 4
SCPU OP 5

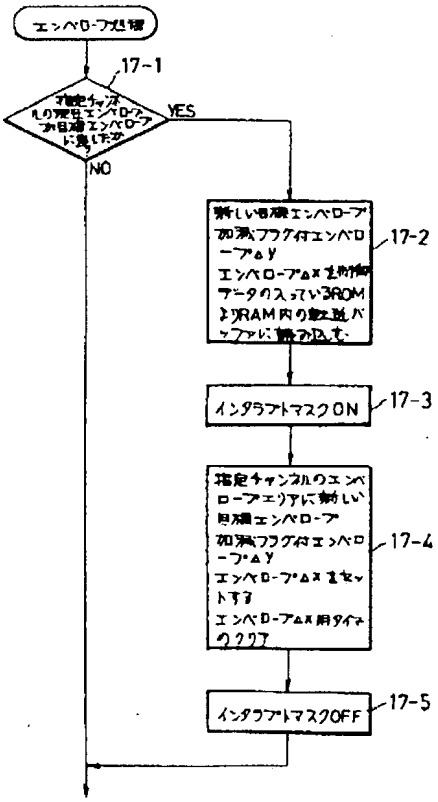
第 14 図


CK2
CK1
T1
SCPU状態
動作
停止
信号A
SCPU処理終了
MCPU
オペレーション
インタラプトOP n
インタラプトOP(n+1)
インタラプトOP(n+2)
インタラプトOP(n+3)
インタラプトOP(n+4)
インタラプトOP(n+5)
インタラプトOP(n+6)
MCPU
オペレーション
アドレス
インタラプトOP(n+1)アドレス
インタラプトOP(n+2)アドレス
インタラプトOP(n+3)アドレス
インタラプトOP(n+4)アドレス
インタラプトOP(n+5)アドレス
インタラプトOP(n+6)アドレス
インタラプトOP(n+7)アドレス
SRT発生
SCPU
オペレーション
SCPU OP
SCPU 最終 OP
SCPU
オペレーション
アドレス
SCPU最終OPアドレス
ダミー OP アドレス

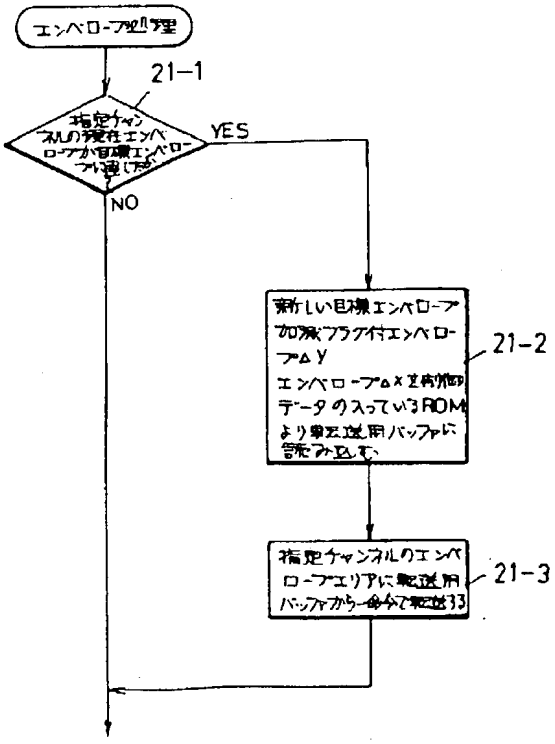
第 15 図



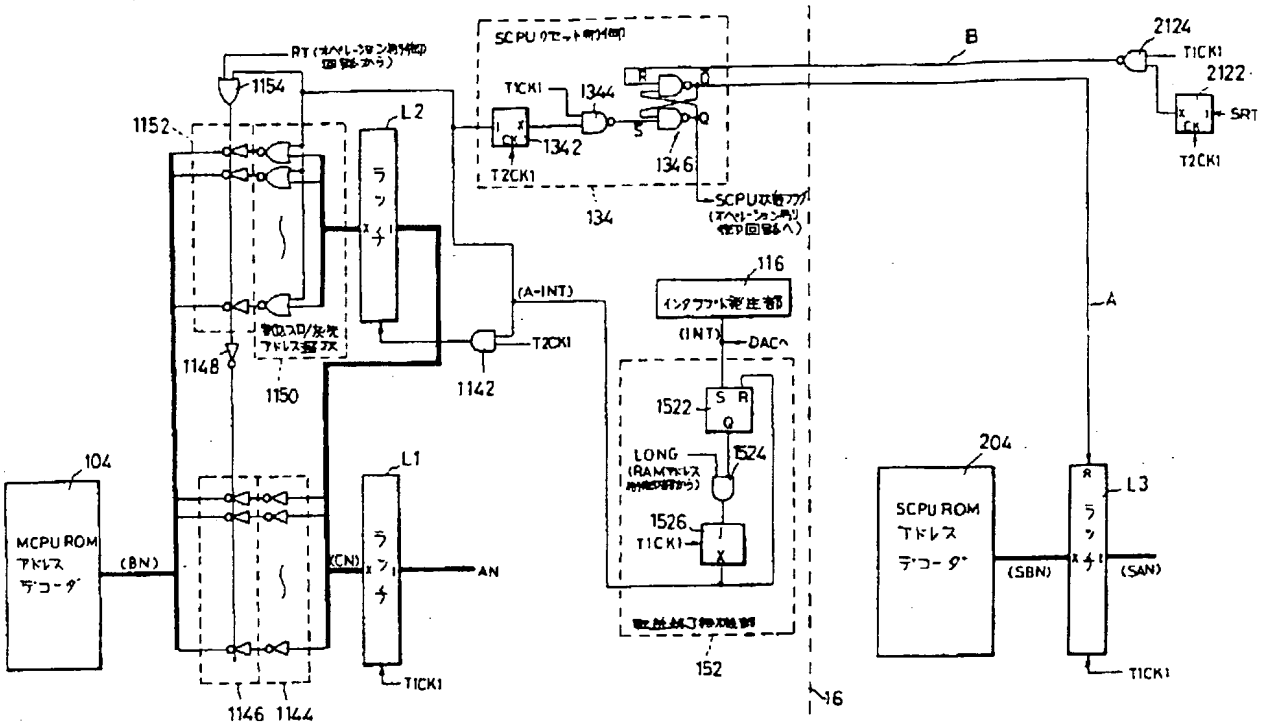
第 16 図



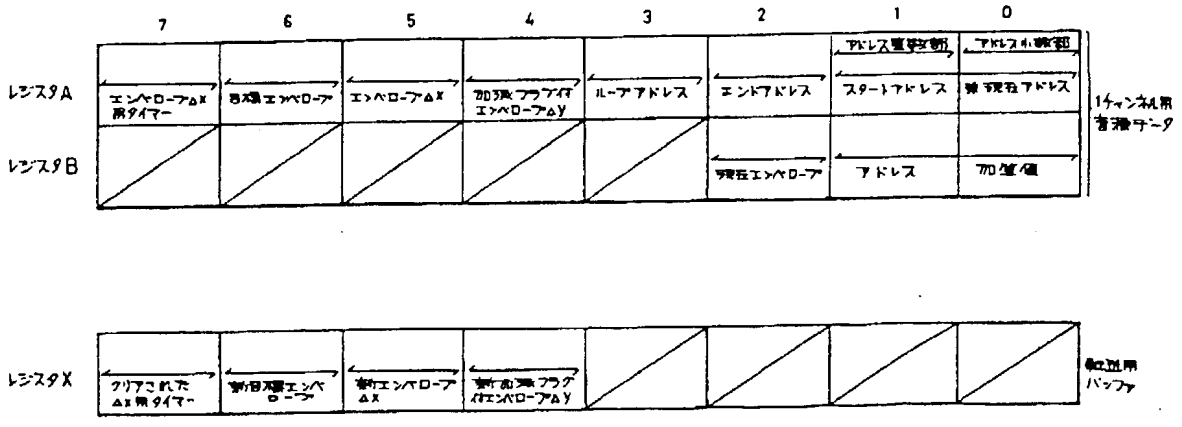
第 17 図



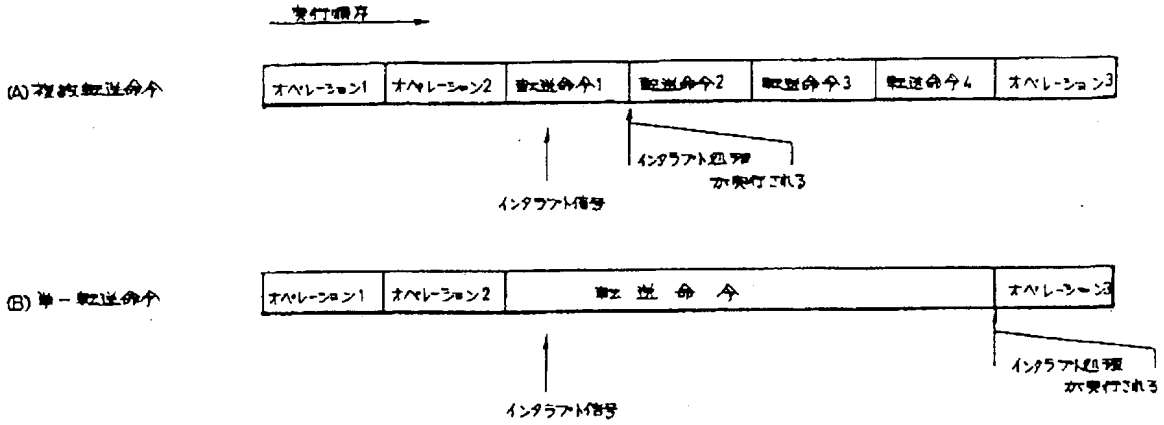
第 21 図



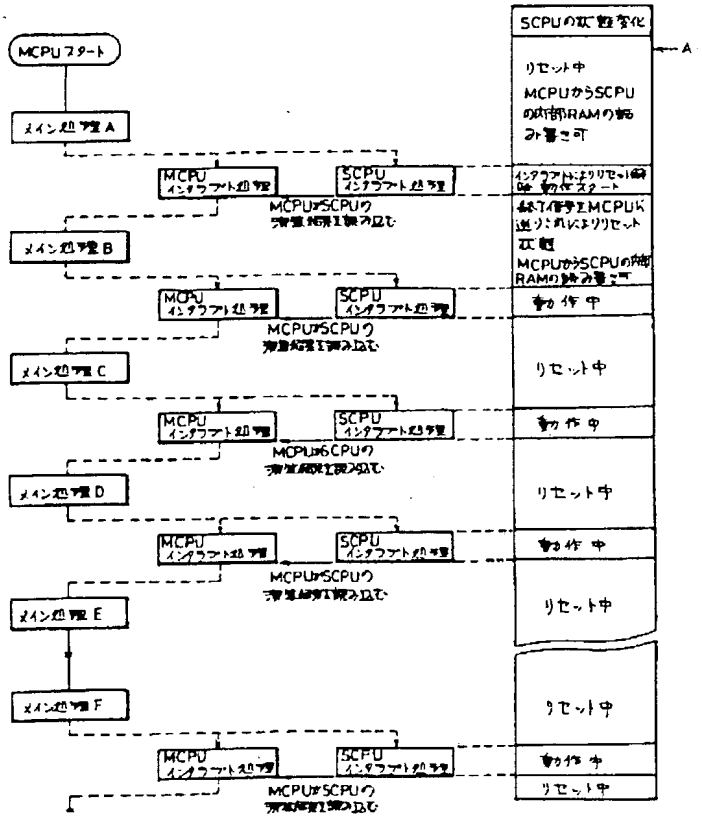
第 18 図



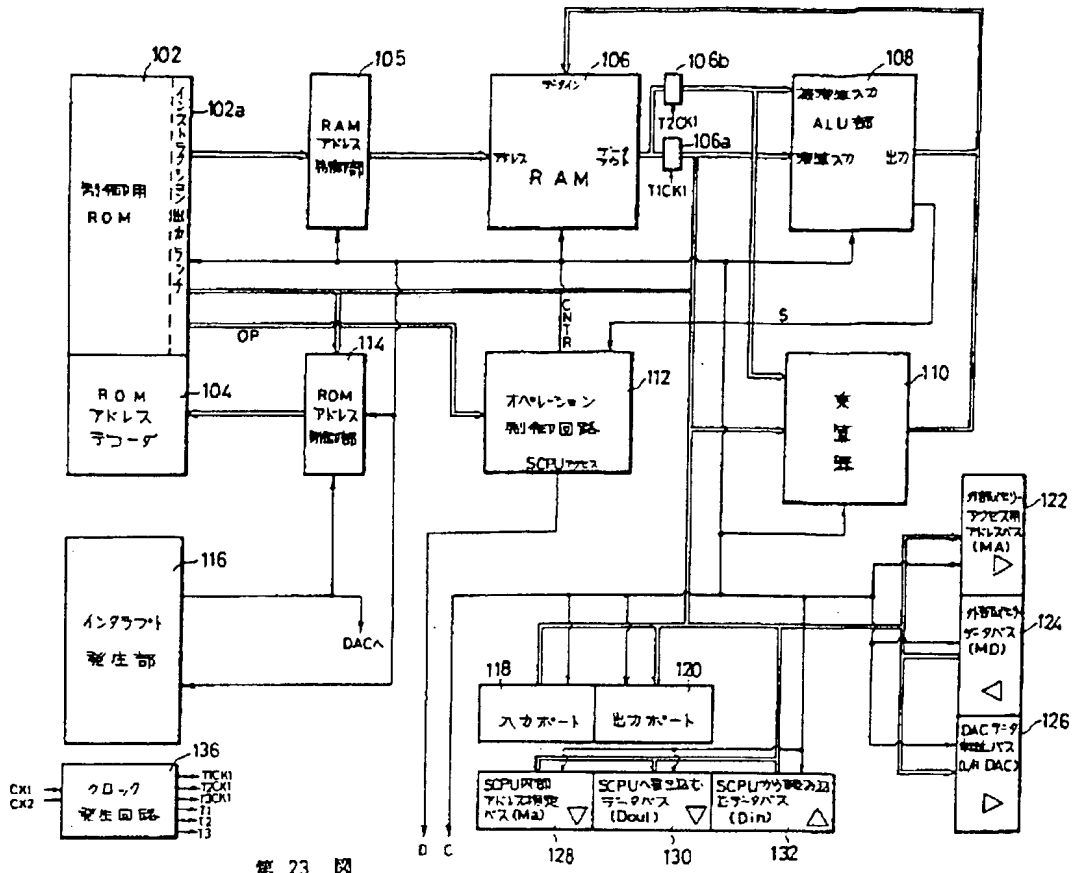
第 19 図
メモリマップ


実行順序
(A)複数転送命令
オペレーション1
オペレーション2
転送命令1
転送命令2
転送命令3
転送命令4
オペレーション3
インタラプト処理
が実行される
インタラプト信号
(B)単一転送命令
オペレーション1
オペレーション2
転送命令
オペレーション3
インタラプト処理
が実行される
インタラプト信号

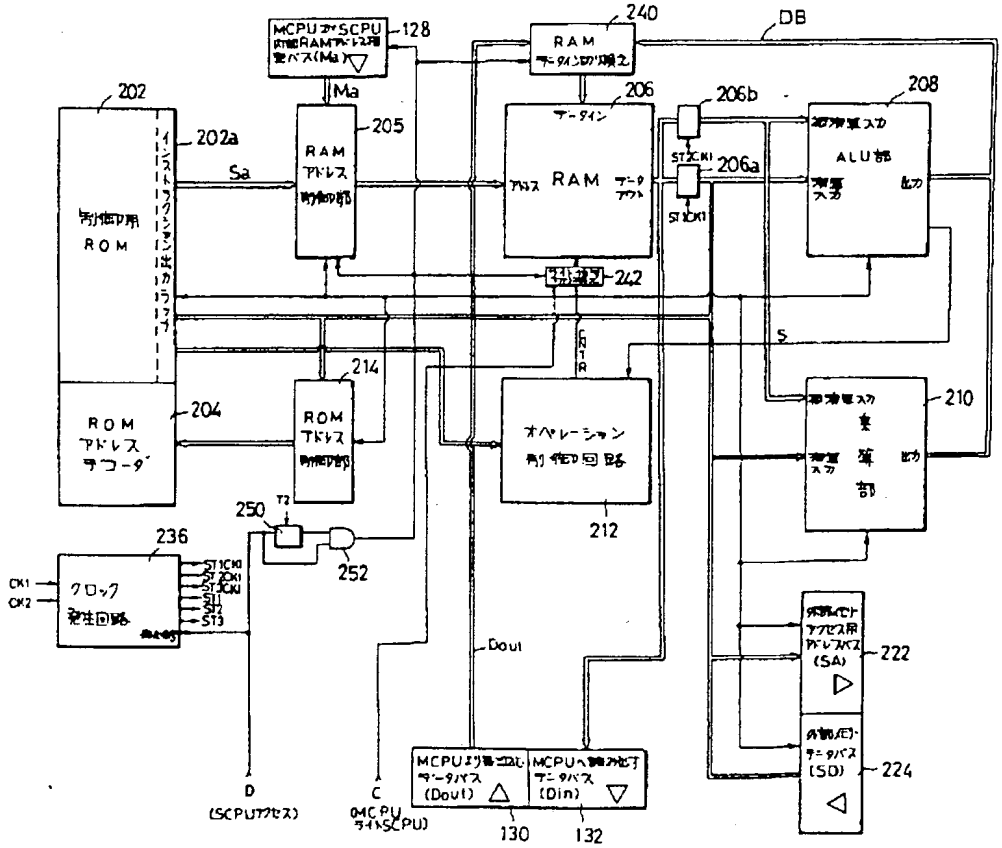
第 20 図


MCPUスタート
メイン処理A
メイン処理B
メイン処理C
メイン処理D
メイン処理E
メイン処理F
MCPU
インタラプト処理
SCPU
インタラプト処理
MCPUからSCPUの
MCPUからSCPUの
MCPUからSCPUの
MCPUからSCPUの
MCPUからSCPUの
SCPUの状態変化
A
リセット中
MCPUからSCPU
の内部RAMの読
み書き可
リセット中
動作中
リセット中
動作中
リセット中
動作中
リセット中
動作中
リセット中

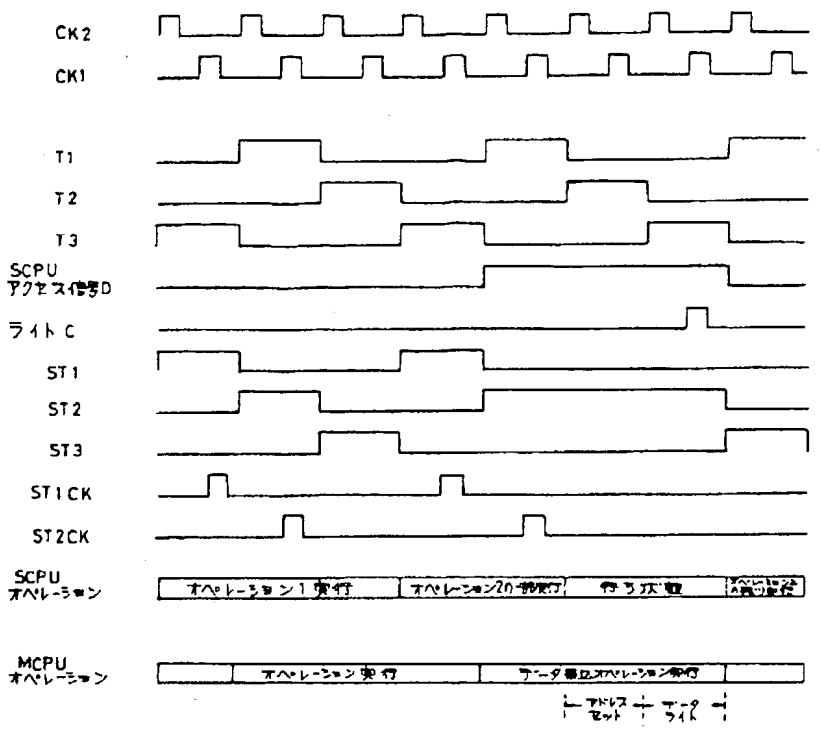
第 22 図



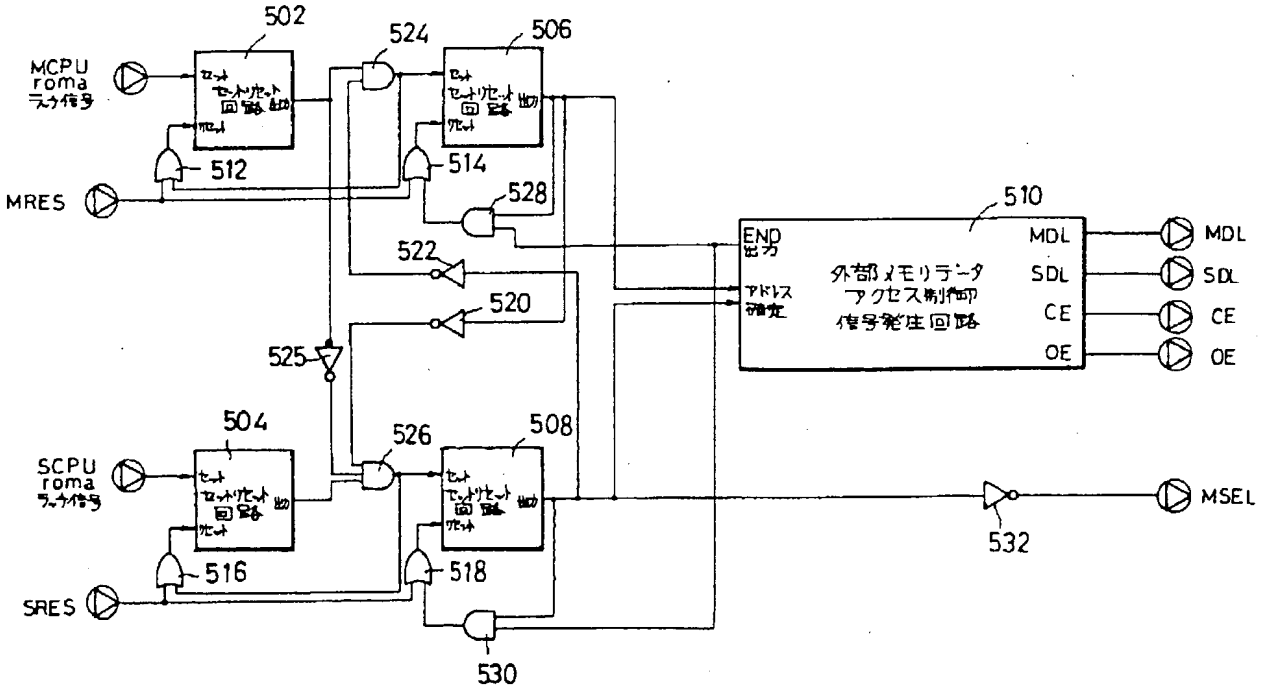
第 23 図
MCPUのブロック図



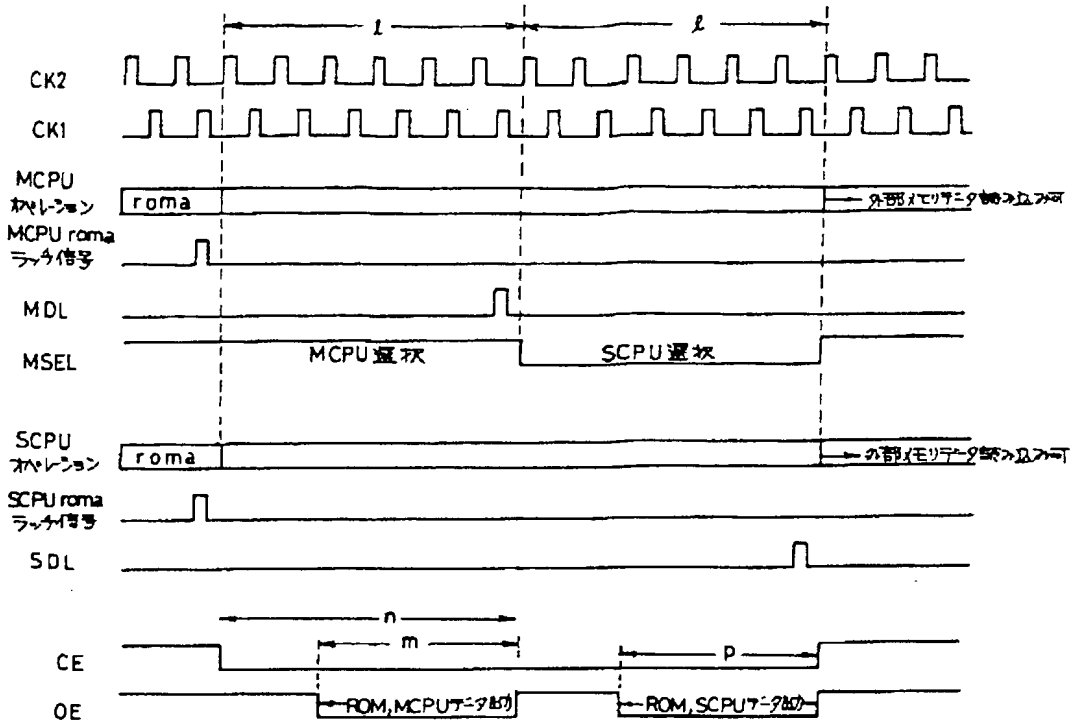
第 24 図
SCPUのブロック図


CK2
CK1
T1
T2
T3
SCPU
ライトC
ST1
ST2
ST3
ST1CK
ST2CK
SCPU
MCPU

第 25 図


502
524
506
MCPU
MRES
512
514
528
522
520
525
504
526
508
SCPU
SRES
516
518
530
510
END
出力
アドレス
確定
外部メモリデータ
アクセス制御
信号発生回路
MDL
SDL
CE
OE
MSEL
532

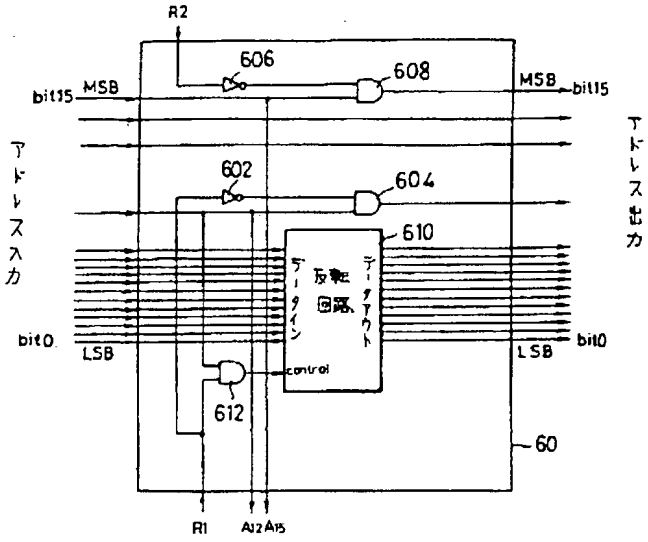
第 26 図



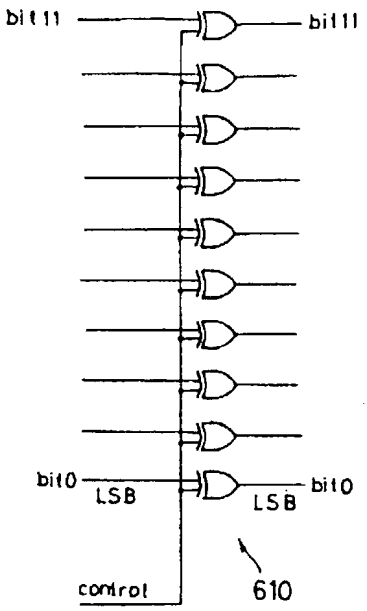
第 27 図

命令名	R1 R2 R3	動作
roma 0	0 0 0	アドレス，データ無変換
roma 1	1 0 0	特殊波形読み出しオペレーション
roma 2	0 1 0	外部ROMデータの一部読み出しオペレーション A15=0 のとき　下位8ビットリード A15=1 のとき　上位8ビットリード
roma 3	0 0 1	ROMデータ　有効データ　12ビット　4ビット CPU読込データ　有効データ上にシフト　0 0 0　12ビット　3ビット

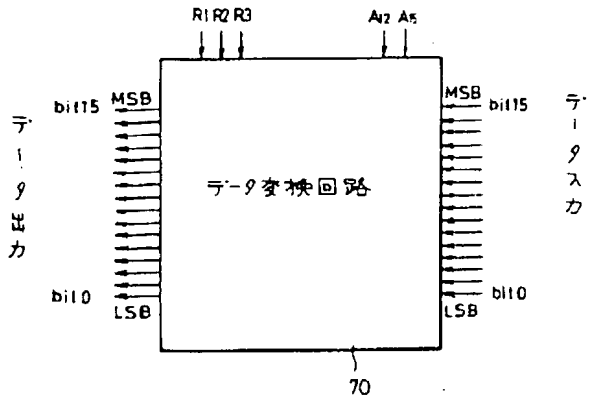
第 28 図



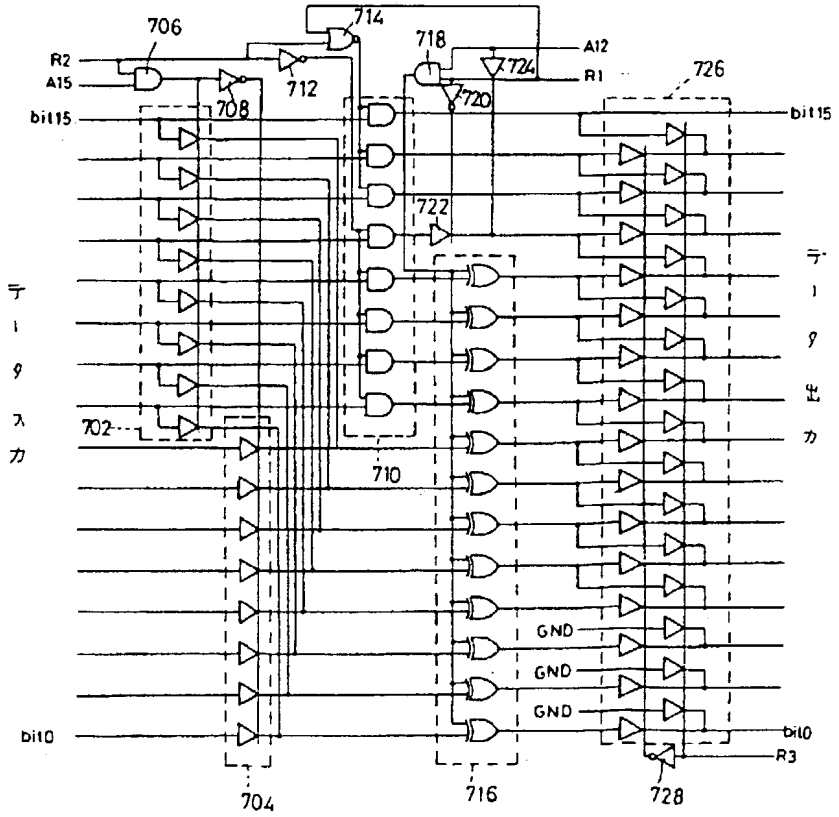
第 29 図



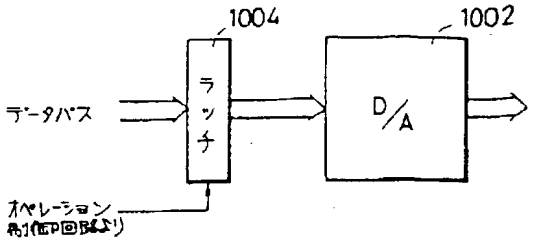
第 30 図



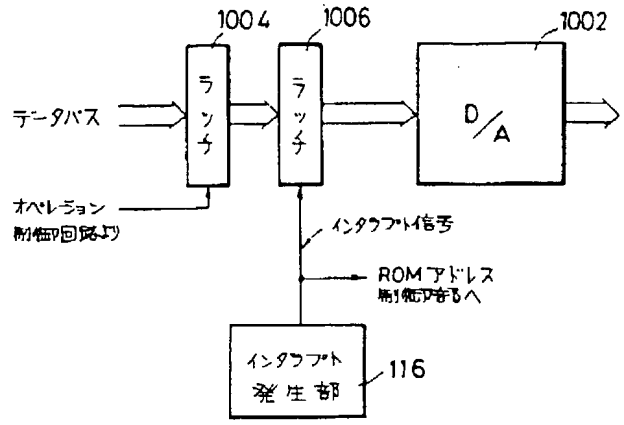
第 31 図
データ変換回路



第 32 図

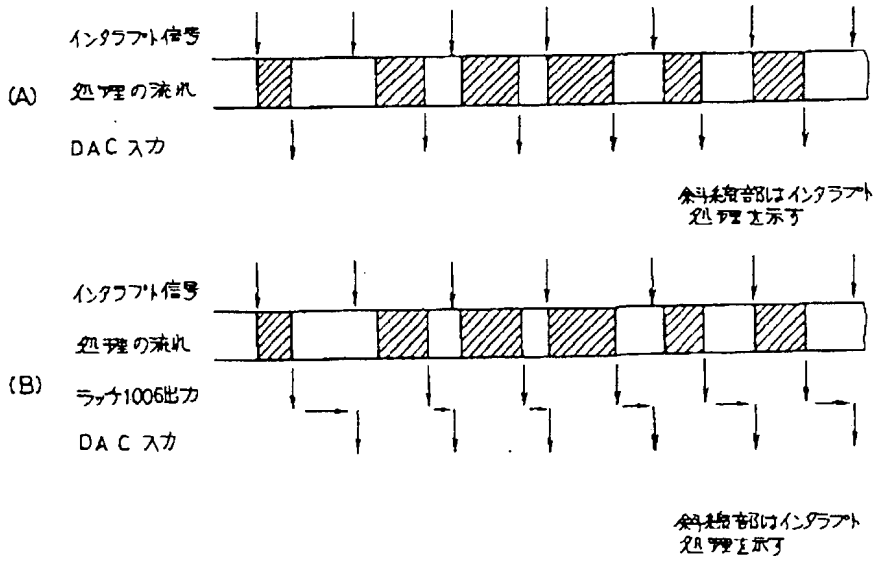


(A) ソフト制御のみの場合



(B) 同期化機能

第 33 図



第 34 図

第1頁の続き

⑫発 明 者 細 田 潤 東京都西多摩郡羽村町栄町3丁目2番1号 カシオ計算機株式会社羽村技術センター内